Mise en place d'interfaces inter-applicatives dans le cadre d'un projet d'implémentation du progiciel de gestion intégré SAP ECC

Benoît Robial

CONSERVATOIRE NATIONAL DES ARTS ET METIERS
CENTRE D'ENSEIGNEMENT DE LYON

MEMOIRE

Présenté en vue d'obtenir le
DIPLOME d'INGENIEUR CNAM
SPECIALITE INFORMATIQUE
OPTION: SYSTEME D'INFORMATION (ISI)

par

ROBIAL Benoît

Mise en place d'interfaces inter-applicatives dans le cadre d'un projet d'implémentation du progiciel de gestion intégré SAP ECC

Soutenu le 13 janvier 2011

Cnam

JURY

PRESIDENT: Christophe PICOULEAU

MEMBRES: Bertrand DAVID, Claude GENIER, Jean-Yves BENEDEYT, Laurent MAYJONADE

Remerciements

Je tiens avant tout à remercier:

Monsieur Laurent MAYJONADE, responsable des intergiciels de la société Givaudan, pour m'avoir fait confiance dès l'issue de mon cursus scolaire classique, pour m'avoir formé, conseillé et permis de rejoindre le projet d'implémentation SAP dès son lancement officiel au mois de septembre 2006.

Monsieur Jean-Yves BENEYDET, chef de l'équipe technique du projet Outlook, pour m'avoir choisi dans son équipe, soutenu et fait confiance tout au long de mon travail à ses côtés. Sous sa responsabilité, ma participation aux travaux tant en équipe qu'en autonomie sur des sujets aux contraintes techniques - et parfois humaines - complexes m'a permis d'acquérir de manière accélérée des connaissances aujourd'hui clés pour mon avenir professionnel.

Ma famille, mes amis, pour leur soutien continu et leur compréhension quant à mon manque de disponibilité durant les périodes les plus intenses de mon cursus au conservatoire national des arts et des métiers.

Résumé

Les applications qui constituent les systèmes informatiques d'aujourd'hui sont le plus en plus complexes et interconnectées. Dans ce contexte, en formant un élément clef d'une solution informatique, un progiciel de gestion intégré est un système dont dépendent la majorité des autres composants. Ainsi remplacer cet élément engendre souvent une multitude de projets complexes, tant dans leurs aspects informatiques que humains.

Edité par la société allemande SAP AG, le progiciel de gestion intégré SAP ECC est de nos jours très présent dans les entreprises industrielles. Au cours des vingt derniers années, de nombreuses sociétés on fait le choix de migrer leur infrastructure informatique vers ce système et se sont retrouvées être confrontées aux mêmes problématiques d'intégration. Durant cette période, des techniques standardisées par SAP sont apparues et une famille d'outils d'intégration spécialisés appelés « intergiciels » ont fortement évolué.

En s'appuyant sur divers cas rencontrés au cours d'un projet d'implémentation du système SAP ECC, ce mémoire illustre et détaille les techniques d'interfaçage permettant de procéder à l'intégration du progiciel dans un écosystème informatique.

Mots clefs: SAP ECC, progiciel de gestion intégré, interfaces, intégration, intergiciel, ALE, IDoc, WebMethods, DataStage.

Abstract

Nowadays, software and informatics systems are becoming more and more complex and interconnected. In this context, an enterprise resource planning system is a component into which most of the other software pieces depend. Consequently, changing this component is always involving multiple complex projects, both in their computing and human considerations.

Built by a German corporation named SAP AG, the SAP ECC enterprise resource planning system is today largely used by the industrial companies. Over the last twenty years, many companies have chosen to migrate their IT infrastructure to this platform and all had to face to the same type of integration problems. During this period, SAP standardized solutions for interfacing systems have emerged and a group of specialized integration tools named as « middleware » became more and more mature.

Relying on various cases encountered during an implementation project of SAP ECC, this thesis documentation illustrates and describes the technical solutions to proceed with the integration of a new enterprise resource planning software in a computing ecosystem.

Keywords: SAP ECC, Enterprise Resource Planning, interfaces, integration, middleware, ALE, IDoc, WebMethods, DataStage.

Sommaire

Introduction

Influencés par des changements de régulation fréquents et des contraintes concurrentielles toujours plus fortes, les processus métiers des entreprises industrielles d'aujourd'hui sont de plus en plus complexes. En conséquence, parce qu'ils forment la base du système d'information de la majorité des entreprises, les progiciels de gestion intégrés sont continuellement adaptés pour répondre aux nouvelles contraintes des organisations métier. Ainsi, le choix de changer de progiciel ouvre toujours des projets complexes et stratégiques pour les entreprises car ils sont l'occasion de procéder à une revue complète de la manière de travailler.

Dans le contexte d'aujourd'hui où les applications et les systèmes informatiques sont de plus en plus interconnectés, un tel projet revient de plus à remplacer la pièce maitresse d'un écosystème par une autre. Les interfaces d'échange de données doivent donc être adaptées au nouveau système, ouvrant ainsi en cascade une multitude de sous-projets d'adaptation d'applications construites avec des technologies diverses.

Travaillant dans le service informatique de la société Givaudan Suisse S.A. depuis l'année 2004, j'ai rejoint le projet d'implémentation du progiciel de gestion intégré SAP ECC dès son démarrage en septembre 2006. Mon rôle fut alors de prendre en charge la responsabilité du suivi et de la réalisation des aspects techniques des interfaces entre le nouveau système et les applications tierces. Ce mémoire synthétise l'expérience et les compétences acquises durant quatre années de travail consacrées à temps complet à la mise en place des communications électroniques entre le progiciel SAP ECC et son écosystème applicatif au sein de Givaudan.

Dans un premier temps, la société Givaudan, le projet Outlook et le progiciel de gestion intégré SAP ECC seront présentés. Les choix des principaux outils et des standards d'intégration sont ensuite détaillés et les technologies seront une à une expliquées. Puis, trois cas d'intégration aux contraintes différentes et représentatives du travail réalisé sur le projet d'implémentation seront étudiés. Enfin, le mémoire se termine par la description de divers outils d'administration et de surveillance qui furent conçus afin de faciliter la gestion des interfaces du nouveau système.

1 La société Givaudan

1.1 Présentation

Givaudan est une entreprise industrielle dont le métier est de concevoir et produire des arômes et des parfums au bénéfice d'entreprises internationales, régionales et locales. Les clients utilisent les solutions ainsi conçues pour fabriquer des produits alimentaires, des boissons, des parfums servant à l'élaboration de produits d'hygiène et d'entretien ou encore à la confection dans le domaine de la parfumerie de luxe. Ainsi, Givaudan est un fournisseur des entreprises vendant leurs fabrications aux consommateurs finaux.

Les principaux secteurs de marché sur lesquels la société opère sont les suivants:

Segment des arômes

- Les boissons: boissons non alcoolisées gazeuses et non gazeuses, jus de fruits, boissons alcoolisées et boissons à préparation instantanée.
- Les produits de confiserie: pâtisseries, friandises, chewing-gums et chocolats.
- Les produits laitiers: glaces, produits laitiers et à préparation instantanée.
- Les produits salés: plats cuisinés, en-cas, soupes, sauces, viandes et volailles.

Avec 2 135 millions de francs Suisse de chiffre d'affaire en 2009, la division des arômes représente 54% des ventes de Givaudan. A travers ce segment, les secteurs des boissons et des produits salés représentent chacun 35% des ventes. Les secteurs des produits laitiers et de la confiserie forment quant à eux respectivement 18% et 12% du chiffre d'affaire de la division.

Segment des parfums:

- Les produits de parfumerie fine: eau de Cologne, parfums pour bases aqueuses ou alcoolisées et autres produits pour le corps, le bain et la maison.
- Les produits fonctionnels: produits pour les textiles et le ménage, désodorisants, soins des cheveux et de la peau, hygiène personnelle.
- Les ingrédients de parfumerie: production et commercialisation tant d'ingrédients pour l'industrie que des bases pour les produits textiles et les soins personnels.

La division des parfums compte un volume de 1 824 millions de francs Suisse de chiffre d'affaire produit sur l'année 2009. Les ventes du segment sont largement dominées par le secteur des produits fonctionnels qui représente 67% du volume contre respectivement 13% et 20% pour les ingrédients de parfumerie et la parfumerie fine.

Avec une part de marchée estimée à 25%, Givaudan est le leader mondial de l'industrie des arômes et des parfums. Les principaux concurrents de la société sont IFF, Firmenich et Symrise.

1.2 Présence dans le monde

A travers 82 sites dont 33 usines de production, Givaudan emploi plus de 8500 personnes. Le siège de la société est situé à Vernier, près de Genève en Suisse.

Comme l'illustre la carte ci-dessous, la société Givaudan est présente sur les six continents majeurs: l'Europe, l'Amérique du nord, l'Amérique Latine, l'Asie, l'Afrique et l'Océanie.

Illustration 1: localisation des sites Givaudan à travers le monde. *Image: givaudan.com*

L'Amérique du nord, l'Europe de l'ouest et le Japon forment des marchés matures qui représentent 62% des ventes du groupe contre 38% pour les marchés en voie de développement constitués de l'Amérique Latine, l'Europe de l'est, l'Afrique et l'Asie.

1.3 Historique

La société Givaudan a été fondée en 1895 par Léon et Xavier Givaudan. Conscient de l'importance d'établir une présence à travers le monde, Givaudan réalisa une première acquisition en 1924 en achetant l'entreprise Burton T. Bush localisée dans le New Jersey aux états unis. En 1948 c'est la société Esrolko S.A. basée à Zurich qui fut achetée, ouvrant pour la première fois à Givaudan le marché des Arômes.
Dans les années 1960, le changement des habitudes de vie avec notamment la démocratisation des vies professionnelles pour les femmes stimula grandement le secteur des arômes. Les foyers

passèrent de moins en moins de temps à cuisiner et augmentèrent par conséquent leur consommation de plats préparés.

Givaudan fut racheté en tant que filiale par le groupe Hoffmann La Roche en 1963. Une année plus tard Hoffmann La Roche acquis l'entreprise Roure, jusqu'alors concurrente de Givaudan. Les deux filiales continuèrent de travailler indépendamment, allant même jusque chacune ouvrir une école de parfumerie qui plus tard furent fusionnées à Paris.

Givaudan étendit ensuite sa présence vers de nouveaux marchés importants tels ceux de l'Asie et de l'Amérique Latine. En 1980 Givaudan acquit Riedel Arom, une entreprise Allemande spécialisée dans les arômes, puis la compagnie acheta Fritzsche, Dodge et Olcott une société américaine réputée dans le domaine des arômes et des parfums dont l'origine date du 18ème siècle. Cette même année, Givaudan et Roure fusionnèrent pour former la compagnie Givaudan-Roure. L'entreprise ainsi constituée fut agrandie en 1997 par l'acquisition de la société américaine Tastemaker, faisant ainsi de Givaudan la plus grande compagnie du monde spécialisée dans l'industrie des arômes.

Le 8 Juin 2000 Givaudan-Roure se sépara du groupe Hoffmann La Roche et conserva uniquement le nom de Givaudan. La société fut alors intégrée à la bourse Suisse sous le code GIVN. Le rachat de FIS, la division arômes de Nestlé fut entériné en 2002. Puis vint en 2003 celui d'IBF, le leader dans le domaine des arômes pour les fromages.

Givaudan annonça le 22 novembre 2006 l'acquisition au groupe ICI d'un concurrent majeur nommé Quest International. Cet achat fut approuvé le 2 mars 2007 par les différentes autorités de régulation des marchés financiers.

Forte de cette histoire faite de multiples fusions et acquisitions, Givaudan est aujourd'hui le leader mondial de l'industrie des arômes et des parfums. La compagnie forme par ailleurs un tiers des parfumeurs et aromaticiens de l'industrie.

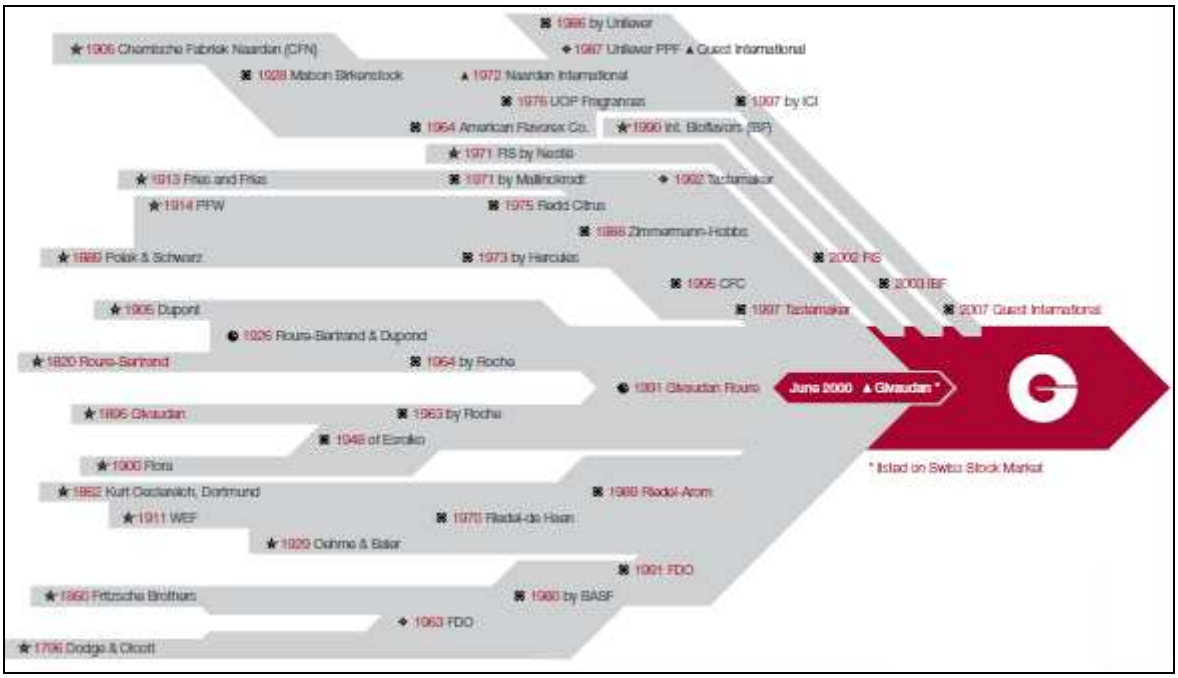


Illustration 2: l'histoire de Givaudan au travers des fusions-acquisitions. Document répliqué en annexe. *Image: givaudan.com*

2 Le projet Outlook

2.1 Objectifs

Le projet Outlook est né d'un effort partagé entre l'organisation informatique et les responsables métier de Givaudan. Les raisons principales étaient de pouvoir subvenir aux demandes croissantes de la chaîne logistique, des fournisseurs et clients, ainsi que de trouver une solution à l'obsolescence de BPCS, le progiciel de gestion intégré - PGI - jusqu'alors utilisé.
Dès lors, les responsables des divisions arômes et parfums ainsi que les divers managers des structures du département informatique se sont regroupés pour procéder à une évaluation des diverses alternatives au PGI en place. A l'issue de cette étude, le progiciel SAP ECC fut déterminé comme le plus à même de répondre aux besoins de l'entreprise.

En implémentant le PGI SAP ECC ainsi que divers autres composants de son écosystème informatique, le projet Outlook a pour but de mener un important travail de transformation des processus métier. Cette mise en place est de plus l'occasion d'harmoniser les systèmes et les données d'une entreprise dont l'histoire est faite de multiples rachats, parmi lesquels celui de Quest International dont l'annonce fut faite seulement trois mois après le démarrage officiel de la phase d'analyse du projet.

Alors que le progiciel BPCS disposait d'une installation dédiée pour chaque site Givaudan, la nouvelle solution se doit de fonctionner sur une unique instance afin d'assurer une meilleure intégration entre les sites et services internes à l'organisation. Par ailleurs, l'acquisition de Quest International a amené un nouveau challenge à l'équipe du projet Outlook: intégrer des sites inconnus, disposant pour chacun de processus et systèmes très différents.

Bien que la mise en place de SAP ECC soit le principal objectif du projet Outlook, il est à noter que l'écosystème SAP à implémenter comporte aussi les composants BI, GTS et SSM. BI, pour « Business Information » est un système d'informatique décisionnelle permettant de réaliser des rapports pour les analyses métier. « Global Trade Services » - GTS - est un nouveau produit SAP dont l'objectif est de gérer des processus du commerce international, tel que la communication électronique avec les douanes et la gestion de divers règlements Européens. Enfin « SAP Solution Manager » - SSM - est un outil central de support permettant d'implémenter, gérer, administrer, superviser et maintenir les systèmes et solutions SAP. Interfacés avec SAP ECC au travers de mécanismes standards qu'il a juste été nécessaire d'activer, ces trois composants ne seront pas abordés au cours de ce mémoire.

2.2 Organisation

Selon les différentes phases, le nombre de personnes mobilisées sur le projet oscille entre 100 et 250. Ces ressources sont pour la moitié issues du personnel Givaudan, incluant majoritairement le site de Genève mais aussi des personnes venant des autres sites Européens, Américains ou Asiatiques. La seconde moitié représente des consultants partenaires dans la progression du projet. Ainsi, la société Accenture fournit un nombre important de ressources clefs à l'organisation et l'entreprise portugaise ROFF met quant à elle à disposition prés de 40 personnes pour former les équipes de développement et la structure de support aux utilisateurs. Le fort caractère international et multiculturel induit par la diversité des membres du projet et des implémentations locales de Givaudan requiert à l'ensemble de l'organisation Outlook de travailler et produire des documents uniquement en langue anglaise.

Sponsorisé par Gilles Andrier, le directeur général de Givaudan, l'évolution du projet est contrôlée par un comité de pilotage formé de quatre hauts responsables métier siégeant aussi au conseil d'administration de l'entreprise. Les structures de gestion et de validation interne comportent quant à elles le chef du projet et divers autres responsables métiers à l'échelle globale de Givaudan.
Pour chaque domaine, des équipes dédiées appelées « streams » ont pour charge de procéder à l'analyse des besoins, à la configuration du système, à l'écriture des spécifications fonctionnelles ainsi qu'a la réalisation des premiers tests. Le découpage fonctionnel de ces équipes suit le modèle des entreprises industrielles. Comportant pour chacun entre 5 et 10 personnes, les streams sont listés ci-dessous:

- **Finance (FIN)**: gestion des flux monétaires, livres de comptes, centres de coûts et autres mécanismes nécessaires aux contrôles financiers.
- **Order to cash (OTC)**: gestion des processus en relation avec les clients: commandes, livraisons, facturations, conditions de prix...
- **Manufacturing (MFG)**: gestion des processus de fabrication mis en œuvre dans les usines Givaudan.
- **Plant maintenance (MPM)**: gestion des processus liés à la maintenance des usines, bâtiments et autres installations de la société.
- **Distribution (DIS)**: gestion des stocks, mouvements de stock et du conditionnement (emballage).
- **Procure to pay (PTP)**: gestion des processus d'approvisionnement couvrant les matières premières mais aussi les fournitures en outillage et les ressources humaines externes.
- **Quality (QUA)**: gestion des processus de contrôle qualité.
- **Compliance (COM)**: gestion des processus de conformité aux différentes réglementations sanitaires des produits fabriqués par les usines.

La responsabilité de l'ensemble des équipes fonctionnelles et la coordination des travaux transverses sont assurés par l'équipe Solution Delivery Management (SDM). La structure Enterprise Architecture & Planning (EAP) gère quant à elle la gouvernance des données maitres (objets clients, produits, formules...) et la prise des décisions avec les organisations extérieures au projet concernant les orientations à long terme.

La structure Technical Delivery Management (TDM) est quant à elle le regroupement sous la même responsabilité de trois équipes: développement, administration SAP et architecture technique. Hormis les ressources propres au centre de développement externalisé, TDM représente 15 personnes qui travaillent à temps plein.

Enfin, des équipes de déploiement temporaires sont mises en place suivant la progression du projet afin de procéder localement à la formation des utilisateurs, à la gestion des besoins spécifiques locaux et à la coordination sur site des activités d'implémentation. La structure de ces dernières est calquée sur le découpage fonctionnel des streams de l'équipe centrale.
Aussi appelée « Kernel Team », l'équipe centrale est basée en Suisse à Petit-Lancy dans des locaux loués spécialement pour l'organisation projet. Les équipes de déploiement sont quant à elles situés localement sur les sites à déployer.

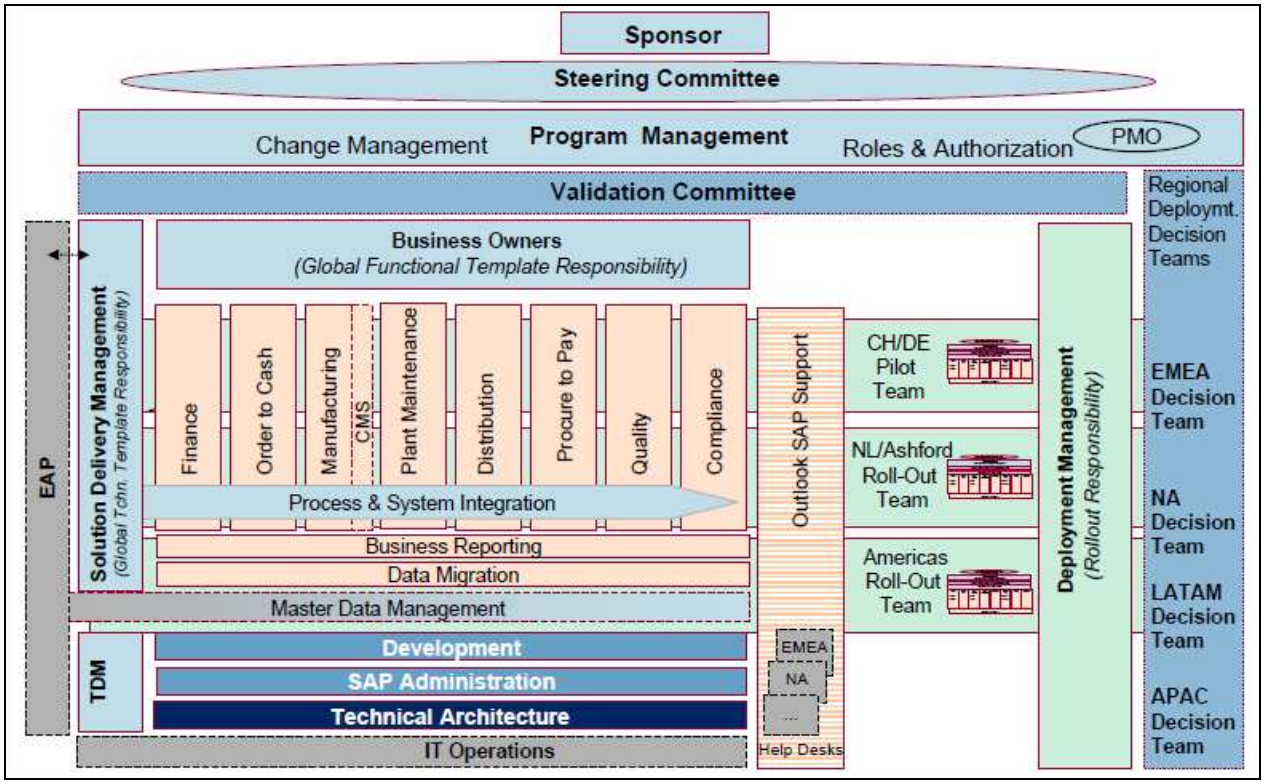


Illustration 3: Organisation du projet. Document répliqué en annexe. *Source: Communication Projet Outlook*

2.3 Planning

Initiées en 2005, les phases de planning et de préparation préliminaire du projet furent terminées au milieu de l'année 2006 et le déclenchement de la réalisation prit place au mois de septembre de cette même année. C'est alors que l'équipe projet fut constituée et réunie dans des nouveaux locaux loués pour l'occasion afin de commencer les travaux par la conception des processus cœurs.
En formant le noyau du nouveau système, les processus cœurs représentent la solution standardisée par Givaudan. Des alternatives à ces processus peuvent être conçues sur demande des sites, néanmoins ces exceptions représentent un coût important qui justifie un accord de la part du comité de validation du projet.

La conception du noyau prit fin au milieu de l'année 2007. Dès lors, la phase d'implémentation des sites pilotes commença. Le premier site pilote qui fut sélectionné est l'usine de Givaudan située en France, à Argenteuil près de Paris. Ce choix était motivé par le besoin de valider les processus cœur sur un site exécutant la plupart des scénarios métier et ayant un volume de production limité. En cas de problème majeur empêchant de livrer les clients, il fut alors prévu de transférer la production au site de Vernier, en Suisse.
Suite à cette première phase de localisation, la mise en production eu lieu au mois de mai 2008. Malgré quelques inévitables problèmes initiaux, l'usine d'Argenteuil fonctionna rapidement sur un système stabilisé. L'implémentation des sites de Givaudan Suisse et Allemagne mobilisa dès lors les effectifs du projet Outlook. Les importants volumes produits par ces usines empêchaient alors tout scénario de repli en cas d'incident. Ces mises en production eurent lieu au mois de novembre 2008 et la stabilisation se prolongea jusqu'au mois de février 2009, clôturant ainsi la phase d'implémentation des sites pilotes du projet.

L'année 2009 fut consacrée premièrement à l'implémentation des sites de l'Angleterre et des Pays-Bas provenant de l'acquisition de Quest International, puis à l'évolution du système pour répondre aux besoins des sites productifs qui avaient accepté de retarder la livraison dans le système SAP de certaines fonctionnalités dont ils disposaient auparavant sur BPCS. Le démarrage des sites acquis eu lieu au mois d'octobre 2009 et leur stabilisation fut terminée au début du mois de novembre de cette même année. Le succès de ces deux premières implémentations de la phase de déploiement permit à Givaudan de capitaliser de la confiance dans la capacité de l'équipe Outlook à déployer sa solution efficacement sur des sites jusqu'alors peu connus.

Au mois de mars 2010, les sites américains de la division des parfums furent à leur tour migrés vers le nouveau système. Avec le Mexique, le Brésil, l'Argentine et la Colombie, c'est l'intégralité des sites de l'Amérique Latine qui furent implémentés au cours des mois de juillet et aout 2010. Enfin, la mise en production de l'usine de Sant-Celoni en Espagne en novembre 2010 conclut une année importante de travail et succès sur le projet Outlook.

L'année 2011 sera aussi riche en challenges. Au mois de mars les six sites américains de la division des arômes devraient être intégralement déployés sur le nouveau système au cours d'une même étape. C'est aussi au cours de cette année que commenceront les travaux d'analyse pour l'implémentation des sites de l'Asie et du Pacifique. Les différences de culture, de langage et d'alphabet devraient conférer à ces derniers déploiements une importante complexité.
La migration des sites Africains de l'Egypte et d'Afrique du sud n'est pour lors pas encore précisément planifiée. Ces démarrages devraient prendre place au plus tôt fin d'année 2011 ou courant 2012 avec les derniers sites Asiatiques.

L'issue du projet est estimée au courant de l'année 2012. Néanmoins, l'implémentation de la version 7.0 de SAP ECC pourrait y ajouter une dernière étape dont la durée serait estimée à une année.
L'illustration ci-dessous représente à un niveau de détail général le planning du projet Outlook.

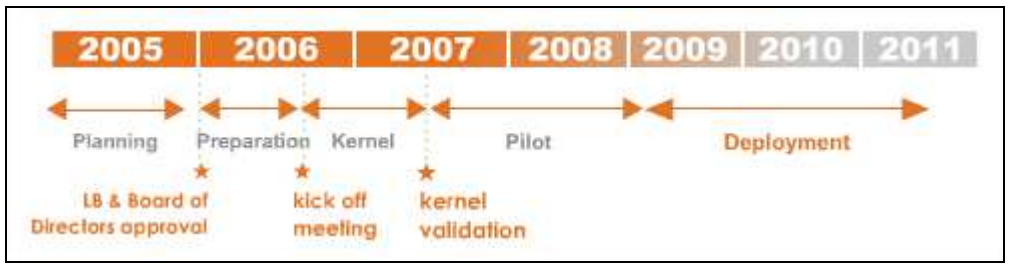


Illustration 4: Planning du projet. *Source: Communication Projet Outlook*

2.4 Cycle de vie des développements

Qu'il s'agisse de nouveaux processus cœurs, d'évolutions ou de spécificités locales, les besoins sont exprimés aux équipes fonctionnels directement par les utilisateurs clefs des équipes métier. Les utilisateurs clefs sont des personnes avec lesquelles le service informatique a noué un contact particulier. Présents dans l'organisation métier de chacun des sites, ils exécutent un premier niveau de support aux utilisateurs, centralisent la gestion des problèmes liés au nouveau système, forment leurs collègues et exécutent des tests dans les divers environnements SAP.

Une fois les besoins exprimés, les équipes fonctionnelles et les utilisateurs clefs formulent dans un document spécialisé une demande de changement. D'un niveau d'abstraction élevé, ce document comporte une analyse d'impact permettant au comité de validation du projet d'accepter ou de rejeter la requête. Lorsqu'ils sont acceptés, les changements sont assignés à une version déterminant la date de livraison dans le système de production.

C'est alors que les analystes des équipes fonctionnelles débutent un travail d'analyse métier détaillé avec les utilisateurs clefs et s'appuient sur les spécialistes de l'équipe technique afin de procéder à une étude des possibilités d'implémentation. A l'issue de cette étape, l'analyste produit une documentation fonctionnelle qui est ensuite soumise à l'équipe technique. L'illustration ci-après représente le cycle de vie de production d'une spécification fonctionnelle.

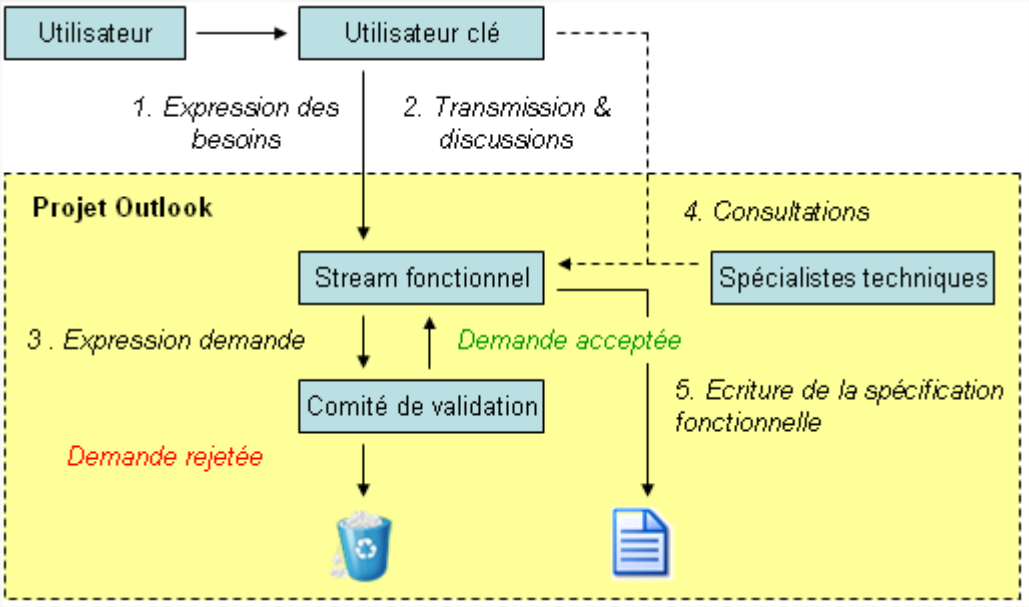


Illustration 5: Cycle de vie de production d'une documentation fonctionnelle.

Après réception des spécifications fonctionnelles, un contrôle qualité est effectué par un spécialiste du domaine concerné: programmes SAP, interfaces, flux de travail - *workflows* - ou encore formulaires imprimés. Une fois les éventuels ajustements réalisés, la documentation est transmise à l'équipe de développement.

Lorsque les développements sont terminés, les analystes procèdent à une phase de test et réclament des corrections ou ajustements directement aux développeurs. Le délivrable est ensuite transporté sur les autres environnements SAP du paysage applicatif afin que les utilisateurs clefs puissent procéder aux tests. La date de livraison en production est déterminée en fonction de la version à laquelle les changements sont assignés.

De par la complexité du domaine, le développement des interfaces poursuit un cycle de vie plus complexe. A la réception des spécifications fonctionnelles, le spécialiste de l'équipe technique initie un travail de coordination entre les développeurs du système SAP, les développeurs externes ou internes au projet spécialisés dans les intergiciels présents chez Givaudan, et les experts des applications tierces avec lesquelles la communication électronique est à mettre en place. Contrairement aux autres développements, le suivi à réaliser dans le cadre des interfaces est complet. Ainsi, ce travail requiert de participer aux tests unitaires, aux analyses de performances, au déploiement de la solution sur les autres environnements du paysage applicatif... Cette différence avec les autres types de livrables est rendue obligatoire par la complexité du domaine. En effet la mise en place d'interfaces requiert des compétences diverses sur des systèmes hétérogènes et aussi une prise de conscience des aspects sécuritaires tant il serait possible qu'une défaillance ait un impact majeur sur la disponibilité des systèmes interfacés.

Le schéma ci-après représente le cycle de vie des développements sur le projet Outlook en fonction du domaine technique concerné.

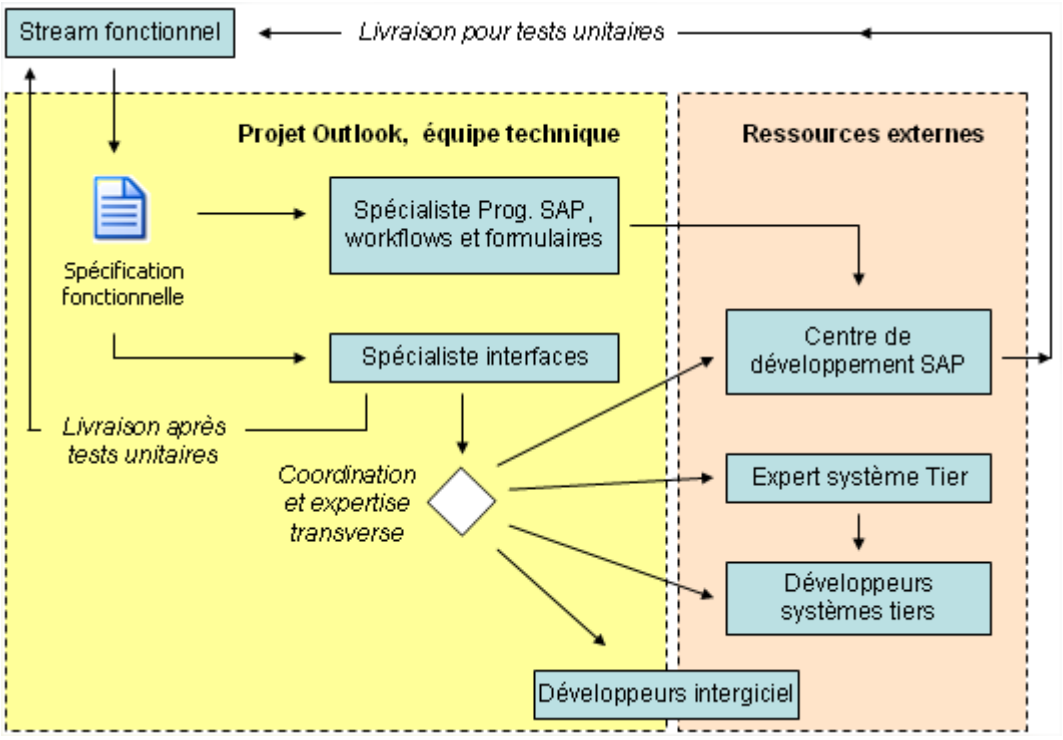


Illustration 6: Cycle de vie des développements sur le projet Outlook.

2.5 Mon rôle

Mon rôle sur le programme Outlook a été de prendre en autonomie la responsabilité de tous les aspects techniques des interfaces avec le nouveau système dans le rôle de spécialiste des interfaces tel que précédemment décrit. Mon domaine d'expertise couvre ainsi SAP, les intergiciels d'intégration et certaines applications tierces que j'ai eues l'occasion d'étudier ou pour lesquelles aucun expert n'était en mesure de m'aider.

Au cours de la première phase du projet j'ai réalisé une étude « as-is » sur les communications entre BPCS et les systèmes tiers existants. Ce travail m'a permis de prendre conscience du futur paysage applicatif de la solution et de connaitre une partie des contraintes auxquelles nous allions avoir à faire face. Dès lors, je me suis engagé dans la définition de nouveaux standards de développement sur le bus d'intégration et sur la configuration de la couche ALE de SAP. Durant la première année du projet, ces documents ont régulièrement été mis à jour en raison de ma compréhension progressive du nouveau système. En parallèle à cette activité, j'ai par ailleurs réalisé de nombreux prototypes afin de vérifier l'efficacité de certains concepts.

Une fois l'étude préliminaire terminée nous sommes entrés dans la phase de conception en commençant par la définition du cœur du nouveau système, c'est à dire les fonctionnalités amenées à être communes à tous les sites. C'est alors qu'un travail plus intensif à débuté. Les équipes fonctionnelles étaient en charge d'écrire les spécifications des interfaces et me délivraient ensuite ces documents afin de procéder à leur validation. Après plusieurs cycles de retouches et une recherche de synergies avec d'autres interfaces existantes, je transmettais les spécifications

aux équipes de développement chargées d'écrire le code nécessaire sur SAP ainsi que sur les intergiciels d'intégration. Les adaptations sur les systèmes tiers étaient quant à elles réalisées par des experts internes à Givaudan avec qui je communiquais fréquemment. Une fois les développements terminés je vérifiais le bon respect des standards et validais le fonctionnement des interfaces à l'aide de quelques tests unitaires avant de livrer le résultat aux équipes fonctionnelles.

Lorsque la définition du noyau de la solution fut terminée et validée via une phase globale de tests, une nouvelle étape de spécification et de développement à immédiatement débuté afin d'adapter le système aux spécificités du premier pays que nous avons eu à implémenter: la France. En parallèle à cette phase de développement, l'équipe de migration de données a commencé à tester ces processus de chargement. Comme nombre de ces processus s'effectuaient au moyen des interfaces, j'ai pris en charge une partie de cette activité. Grâce à cette tâche j'ai pu franchir un nouveau palier dans ma compréhension du système car cela fut l'opportunité d'analyser le comportement de SAP lors de conditions de charge extrêmes. Fort de ces expériences, nous avons pu ajuster la configuration et le développement des interfaces afin de s'assurer que leur fonctionnement en conditions intenses ne puisse pas rompre la disponibilité du système, ni affecter les utilisateurs.

Une fois les développements spécifiques à la France terminés et après de multiples séries de tests pour valider notre préparation, nous avons mis en production notre solution. Après une inévitable phase de stabilisation une nouvelle ère s'est ouverte: nous sommes passés en mode « projet-productif ». Dès lors de nombreuses tâches d'investigation m'ont été confiées afin de comprendre les problèmes, proposer des solutions et parfois les spécifier avant de pouvoir procéder rapidement à leur développement.

Mon travail sur la construction des interfaces ne fut pas interrompu durant cette période, le prochain objectif étant l'implémentation combinée des sites Suisses et Allemands, bien plus critiques pour les activités de Givaudan sur le marché Européen. Les développements et les processus de migration étaient devenus critiques car nous devions alors nous assurer de ne pas perturber les utilisateurs ou apporter des régressions. Etant donné le turn-over évident des consultants et donc la fuite de connaissances inévitable dans ce type de projet, mes connaissances transverses acquises au fil des différentes étapes ont été mises à profit pour valider les modifications et opérations techniques demandées sur l'environnement de production.

En raison des contraintes de confidentialité sur les processus critiques de Givaudan mais aussi dans le but de conserver l'intérêt à la lecture de ce mémoire, l'ensemble des interfaces produites au cours du projet - *plus de 60* - ne sera pas abordé. Seuls trois des cas les plus représentatifs des contraintes d'intégration seront détaillés dans les chapitres ci-après.

3 Le progiciel SAP ECC

3.1 Présentation

SAP ECC est un progiciel de gestion intégré édité par la société SAP AG dont le siège se trouve à Walldorf, en Allemagne. SAP est l'acronyme de « Systems, Applications and Products in data processing », tandis que ECC signifie « Enterprise Central Component ».
La société SAP AG est le plus grand concepteur de logiciel en Europe. Employant près de 52 000 personnes, l'entreprise se positionne à la quatrième place des plus grands éditeurs informatique du monde, situé juste derrière Microsoft, IBM et Oracle. La part de marché estimée à son produit phare SAP ECC est aujourd'hui de 60%, justifiant ainsi la confiance accordée par ses clients et plus particulièrement ceux du secteur industriel.

La première version du produit est née en 1973 sous le nom de SAP R/1. Fonctionnant à l'époque sur des machines de type mainframe, le système était spécialisé dans la gestion des traitements financiers. Cette version servit de base pour le développement d'une multitude de nouveaux modules métier. SAP R/2, la seconde version majeure du produit fut quant à elle proposée en version stabilisée à partir de 1981.
En 1992 SAP mit à disposition R/3, un système abandonnant pour la première fois l'architecture mainframe au profit du client-serveur. Cette version connue un succès majeur et de multiples évolutions avant d'être remplacée au mois d'aout 2005 par un nouveau système sous le nom SAP ECC 5.0. En juin 2006, le système évolua à nouveau avec la mise à disposition de SAP ECC 6.0, la version encore la plus actuelle aujourd'hui. C'est cette dernière qui fut installée chez Givaudan. ECC 7.0, la prochaine version du célèbre progiciel de gestion intégré est aujourd'hui toujours secrète, SAP AG communicant très peu à ce sujet. Elle ne devrait pas être mise à disposition avant le courant de l'année 2012.

3.2 Modules

Le PGI SAP ECC 6.0 est constitué d'une multitude de modules fonctionnels. Ces derniers sont répartis en trois familles dont les principaux sont listés ci dessous.

Modules logistiques

- Material Management (MM): gestion des produits, processus liés aux achats, factures et aux mouvements de stocks.
- Production Planning (PP): gestion de la production, planification, calcul des besoins, gestion des capacités et calcul des coûts de revient.
- Sales and Distribution (SD): administration des ventes, gestion des appels d'offre, des offres et contrats, commandes clients, conditions de prix, facturation, expédition, livraisons et gestion des remises.

- Project System (PS): module transverse. Structuration des projets, suivi et coûts, planning et calendrier.
- Quality Management (QM): gestion de la qualité, planification, plans d'inspection, contrôles, certificats et gestion des réclamations.
- Plant Maintenance (PM): gestion de la maintenance, description du référentiel, maintenance préventive et curative, demandes d'intervention, traitement des ordres, historiques, confirmations d'achèvement, coûts et re-valorisation des articles.
- Customer Services (CS): gestion des services clients - extension au module PM.
- Product Life-Cycle Management (PLM): gestion des cycles de vie des produits, gestion de l'environnement, des risques et des données liées à l'hygiène.
- Warehouse Management (WM): gestion globale des entrepôts.
- Handling Unit Management (HUM): gestion des unités de manutention.

Parmi ces derniers, les modules logistiques implémentés sur l'installation SAP ECC de Givaudan sont MM, PP, SD, QM, PM, WM et HUM.

Modules de gestion comptable

- Financial (FI): écriture des ventes et achats, gestion des placements et de la trésorerie, comptabilité générale, client, fournisseur, budgétaire, bancaire et immobilisations.
- Controlling (CO): contrôle de gestion, comptabilité des natures comptables, gestion des activités, ratios statistiques, comptabilité analytique, ordres et projets de frais.
- Controlling Product Costing (CO-PC): calcul des coûts de revient par produit, calcul analytique des supports de coûts.
- Controlling Profitability Analysis (CO-PA): compte de résultat et analyse par segment.
- Treasury (TR): gestion des flux de trésorerie et des paiements.
- Investiments Management (IM): gestion des investissements financiers

Les modules de gestion comptable utilisés sur le système mis en place par le projet Outlook sont les modules FI, CO, CO-PC, CO-PA et TR.

Modules de gestion des ressources humaines

- Personnel Administration (PA): gestion des employés, de la rémunération, primes d'intéressement, suivi du temps de travail, suivi des frais de déplacement.
- Personnel Development (PD): gestion des compétences, suivi des carrières, planification des réservations, gestion des séminaires.
- Personnel Development (PB): gestion du recrutement.
- Employee & Manager Self Service (ESS & MSS): accès intranet pour employés et managers.

Les modules de gestion des ressources humaines ne sont pas utilisés sur l'installation Outlook de SAP ECC. Néanmoins Givaudan dispose d'une installation séparée du progiciel, communément

appelé SAP HR (Human Ressources) sur laquelle certains de ces modules sont actifs. Cette installation est entièrement gérée et hébergé par un partenaire externe spécialisé afin d'éviter les risques évidents de confidentialité. L'accès à ce système par les employés de Givaudan ne faisant pas partie de l'organisation des ressources humaines pourrait en effet leur permettre de consulter des informations sensibles.

3.3 Aspects technologiques

3.3.1 Architecture

Les utilisateurs accèdent aux systèmes SAP à travers le SAP GUI, une application installée sur leurs ordinateurs. SAP GUI est semblable à un navigateur web, le logiciel envoie des requêtes au système SAP, puis réceptionne des données similaires à des pages pour procéder à l'affichage des résultats.
Lors de l'initialisation de la communication avec SAP, un mécanisme nommé portier (Gateway) dirige l'utilisateur vers un serveur d'applications SAP qui sera chargé de procéder à ses requêtes. Ce dernier communique ensuite avec le système de gestion de bases de données (SGBD) afin de consulter ou écrire des données dans la base de données en fonction des actions de l'utilisateur.
Une architecture de système SAP courante consiste à disposer de plusieurs serveurs d'applications SAP afin de pouvoir répondre à un nombre important de requêtes et d'utilisateurs. Dans ce scénario, la coordination entre les serveurs est assurée par un serveur d'applications particulier qui sera configuré de manière à se comporter en instance centrale. Unique à l'installation, l'instance centrale recevra à travers un serveur de messages des demandes de blocages et de déblocages des données afin de procéder à la coordination nécessaire pour gérer les contraintes de concurrence entre les accès. L'illustration suivante représente de manière simplifiée un système SAP disposant d'une telle architecture.

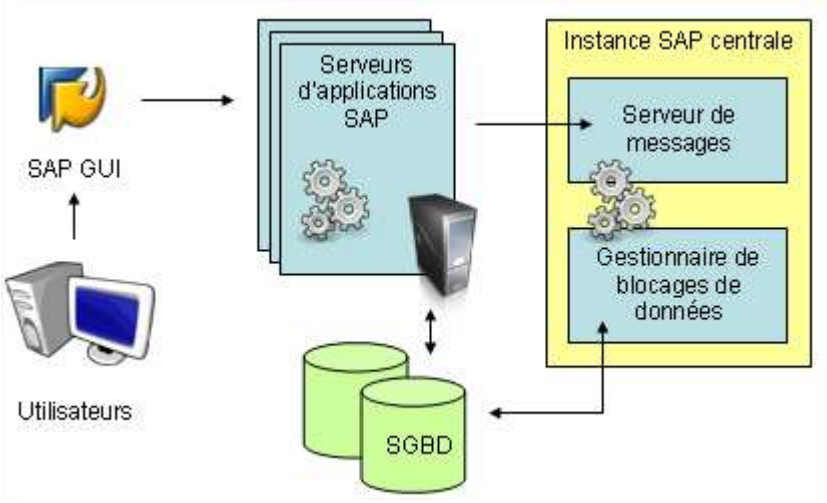


Illustration 7: Exemple d'architecture d'un système SAP.

Au travers d'un mécanisme d'abstraction expliqué dans le chapitre suivant, plusieurs types de SGBD peuvent être choisis pour une installation SAP. Oracle est le SGBD le plus communément sélectionné.

3.3.2 Programmation

La plupart des programmes SAP sont développés dans le langage ABAP. Acronyme de « Advanced Business Application Programming », ABAP est un dérivé du langage COBOL qui permet de développer tant de manière procédurale qu'en orienté-objet. Bien que lors de sa conception le langage avait pour objectif de pouvoir être utilisé par n'importe quel utilisateur, il s'est avéré avec l'expérience que seuls les programmeurs expérimentés pouvaient réaliser des programmes et des rapports.

Tel Java avec sa machine virtuelle (JVM), ABAP est un langage interprété. Les requêtes à la base de données sous jacente sont par ailleurs écrites dans un langage SQL spécifique appelé ABAP SQL. Un tel procédé a pour avantage de conserver une abstraction totale entre les développements réalisés sur la plateforme SAP et le type de SGBD sélectionné. Néanmoins, cette abstraction réduit la puissance du langage SQL et ne permet pas de réaliser de puissantes optimisations dans les accès aux données.

Les développements de nouveaux programmes, tables, interfaces ou autres ont pour particularité de nécessiter de procéder au nommage chaque « objet » en commençant par la lettre Z. Cette obligation propre à SAP s'avère très utile pour distinguer les développements personnalisés créés par les entreprises des développements standards réalisés par SAP AG.

Exemples:

- RBDMIDOC est un objet standard (programme SAP)
- ZGL_SALES_PRG n'est pas un objet standard

3.3.3 Transactions

La navigation dans le progiciel SAP se fait au moyen de transactions. Les transactions sont des codes alphanumériques qui une fois entrés à l'intérieur du champ de saisie situé en haut à gauche de la fenêtre SAP, permettent d'accéder à des programmes. Nombre de programmes techniques ne disposent pas de transaction dédiée, ils peuvent dans ce cas être accédés à travers SE38, la transaction permettant d'exécuter les programmes.

Tels les développements, les codes des transactions personnalisées doivent débuter par la lettre Z.

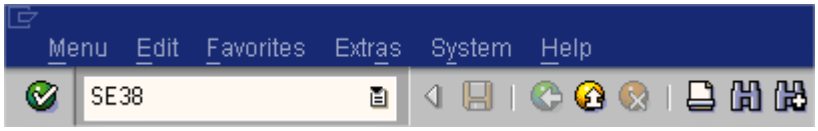


Illustration 8: Zone de saisie des transactions.

4 Outils et standards d'intégration

Les standards et outils d'intégration à utiliser furent déterminés au cours de la première phase du projet. Certains de ces choix ont été repris des technologies et usages déjà en place chez Givaudan, puis adaptés à une utilisation dans le nouveau contexte.

4.1 Les différents types d'interfaces

4.1.1 Les interfaces synchrones

Les interfaces de type synchrone permettent de faire communiquer directement des applications entre elles. L'émetteur initie une communication avec le récepteur en envoyant des éventuels paramètres, puis ce dernier répond immédiatement au travers de la même liaison réseau. Si l'application réceptrice est indisponible, alors la communication est perdue.

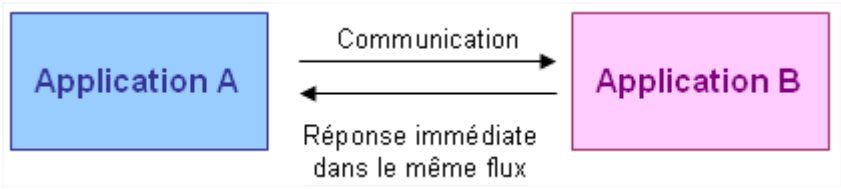


Illustration 9: Communication synchrone.

Bien que permettant d'obtenir une réponse immédiate, il fut décidé de limiter ce type d'interfaces aux uniques cas où les utilisateurs attendent une réponse immédiate, sans pouvoir accepter un mécanisme asynchrone rapide de type requête-réponse s'exécutant en moins de une minute. Les raisons ayant motivé ce choix sont les suivantes:

- L'indisponibilité de l'application réceptrice engendre la perte de la communication.
- Développements, gestion des erreurs et traçage des échanges complexes.
- Pour des raisons de sécurité et de stabilité, il fut décidé de limiter les accès direct à SAP par des applications tierces. L'utilisation d'intergiciels est préférée.

La seule implémentation technique acceptée pour la mise en place d'interfaces synchrones est l'usage de services web. Les services web (communément appelés web-services) permettent au travers d'un protocole standardisé de s'assurer de la pérennité des développements en s'affranchissant des contraintes liées aux différences technologiques entre les applications.

4.1.2 Les interfaces asynchrones pseudo temps-réel

L'intérêt principal des interfaces asynchrones pseudo temps-réel est de permettre des échanges rapides entre les applications tout en s'affranchissant des contraintes de performance et de disponibilité. Dans ce scénario, l'application qui initie la communication n'attend pas une réponse immédiate de la part du récepteur.
Les implémentations de ce type d'interfaces sont réalisées au travers d'une plate forme spécialisée de la famille des intergiciels appelée EAI, acronyme de « Enterprise Application Integration ». Tel qu'illustré sur le schéma ci-après, les EAI sont reçoivent des messages de la part des

applications souhaitant émettre des informations, puis les diffusent aux récepteurs. En cas d'indisponibilité du récepteur ou de performances dégradées, alors les messages sont conservés dans des files d'attente puis remis au destinataire dès que possible.

Illustration 10: Communication asynchrone au travers d'un intergiciel EAI.

Les EAI sont capables de faire communiquer entre elles des applications fonctionnant sur des technologies très différentes. Situé en intermédiaire de chaque échange, l'EAI constitue par ailleurs un point idéal pour superviser et administrer le fonctionnement des interfaces. La durée de latence minimum acceptée est de 30 secondes pour un échange monodirectionnel, ce qui se trouve être satisfaisant dans la plupart des cas d'intégration.

La mise en place de communications au travers d'un EAI constitue la solution préconisée pour la conception des interfaces du projet Outlook. Néanmoins, les EAI sont optimisés pour l'exécution rapide d'échanges de données de faible volume et sont donc entièrement inadaptés pour réaliser des transferts massifs. De tels processus risqueraient de rompre la disponibilité de l'intergiciel.
La plate forme EAI utilisée chez Givaudan est WebMethods, un produit édité par la société Allemande Software AG. Ce dernier est décrit plus loin dans ce chapitre.

4.1.3 Les interfaces batch

A l'inverse des interfaces asynchrones pseudo temps-réel, les interfaces batch sont optimisées pour réaliser des échanges massifs de données, lorsque le temps de latence accepté est supérieur à 15 minutes. Les implémentations de ces interfaces sont réalisées au travers d'une famille d'intergiciels spécifiques appelée ETL, acronyme de « Extract, Transform & Load ». Tel qu'illustré ci-dessous, les multiples tâches d'une interface de type batch sont contrôlées par un ordonnanceur.

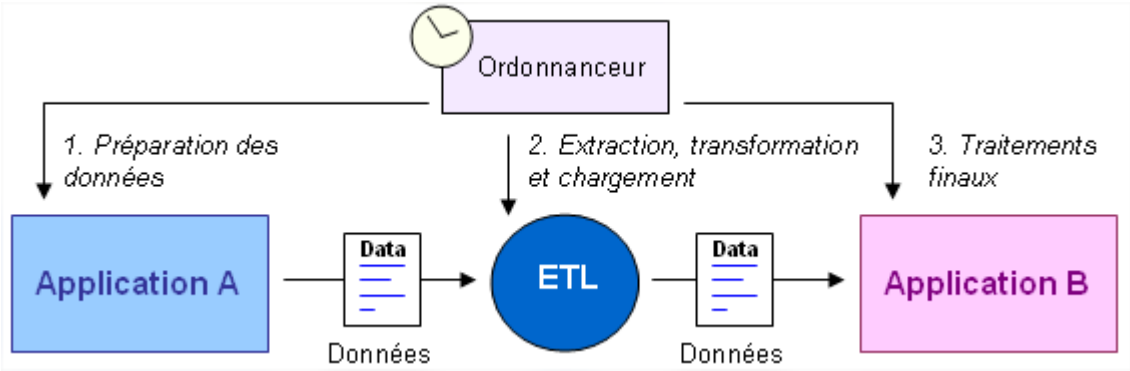


Illustration 11: Communication batch au travers d'un intergiciel ETL.

Dans cet exemple, la première étape de l'ordonnanceur est de déclencher un processus de préparation des données à extraire sur l'application émettrice. Cette étape n'est pas toujours nécessaire car il arrive en effet que l'ETL accède directement aux tables de données de l'application. L'ordonnanceur déclenche ensuite sur l'ETL le programme d'exécution de l'interface. Ce dernier lit les données, les transforme si besoin, puis les transmet à l'application réceptrice. Enfin en cas de besoin, l'ordonnanceur peut déclencher des traitements sur le système récepteur afin d'exécuter des tâches d'intégration post-transfert.

Lors de la définition des standards technologiques du projet Outlook, il fut décidé d'utiliser les interfaces batch uniquement lorsque les conditions suivantes sont réunies:

- Déclenchement sur contrainte horaire ou à la fin d'un autre traitement contrôlé par l'ordonnanceur. Pas de déclenchement sur action utilisateur.
- Transferts de données volumineux sans calcul de delta: l'ensemble des données est envoyé sans réduire uniquement aux informations modifiées dans le système depuis le dernier transfert.
- Latence supérieure à 15 minutes.

L'intergiciel ETL utilisé chez Givaudan est DataStage. Ce produit édité par IBM est décrit plus loin dans cette section. L'ordonnanceur des tâches est ControlM, édité par la société américaine BMC Software.

4.1.4 Les interfaces B2B

Les interfaces B2B, acronyme de « Business to Business » ont pour objectif de réaliser des échanges entre une application interne à la société et le système informatique d'un partenaire externe, tel qu'un client, un fournisseur, une banque ou un transporteur. Le principe est similaire à une interface de type asynchrone temps-réel auquel l'application émettrice ou réceptrice est remplacée par une plateforme spécialisée dans les échanges B2B. L'intergiciel EAI se trouve ainsi connecté au système B2B tel qu'illustré sur le schéma ci-dessous.

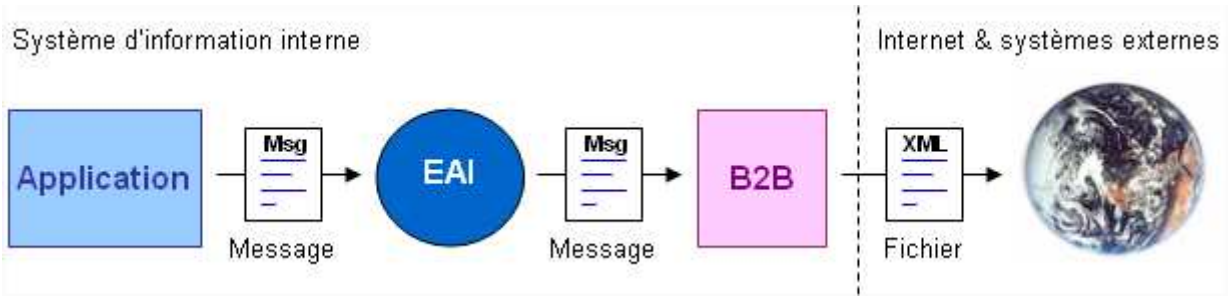


Illustration 12: Communication B2B d'un point de vue émetteur.

Les échanges avec le système externe peuvent être réalisés en utilisant divers protocoles et formats. La solution préférée par Givaudan pour l'implémentation de ces interfaces est d'utiliser le format XML au travers du protocole HTTPS. Néanmoins les contraintes d'intégration des partenaires externes requièrent souvent de la flexibilité sur les règles de construction technique. La plateforme B2B utilisée chez Givaudan est intégrée à l'intergiciel WebMethods.

4.1.5 Tableau récapitulatif

Le tableau suivant dresse le récapitulatif des types d'interfaces et du cadre d'utilisation dans lequel chacun doit être utilisé.

Type d'interface	Latence	Volume par message	Cas d'utilisation
Synchrone	0 seconde (immédiat)	< 1 Méga octet	Uniquement lorsqu'asynchrone temps-réel ne peut pas être utilisé pour des contraintes de latences.
Asynchrone temps-réel	30 secondes minimum	< 1 Méga octet	Tous les cas pour lesquels la latence et le volume sont conformes.
Batch	15 minutes minimum	Tous volumes	Echanges massifs d'informations non limité aux seules données changées.
B2B	Idem asynchrone temps-réel, 30 secondes minimum	< 1 Méga octet	Tous les échanges avec des partenaires externes.

Tableau I: Récapitulatif des types d'interfaces.

4.2 Les intergiciels

4.2.1 L'EAI WebMethods

WebMethods est une plateforme d'intégration EAI et B2B conçue en 1996, puis rachetée en 2007 par la société Allemande Software AG. Le produit est constitué de deux principaux composants:

- Un serveur de messages, aussi nommé message broker ou bus d'intégration. Il s'agit du cœur du concept EAI.
- Un ou plusieurs serveurs d'intégration chargés d'exécuter les développements et de communiquer avec les applications au travers de différents types de connecteurs.

Le schéma ci-dessous illustre en détail l'architecture d'une interface faisant communiquer une application fonctionnant sur un système Oracle avec le système SAP ECC au travers de l'EAI WebMethods en utilisant un message de format nommé « Commande ».

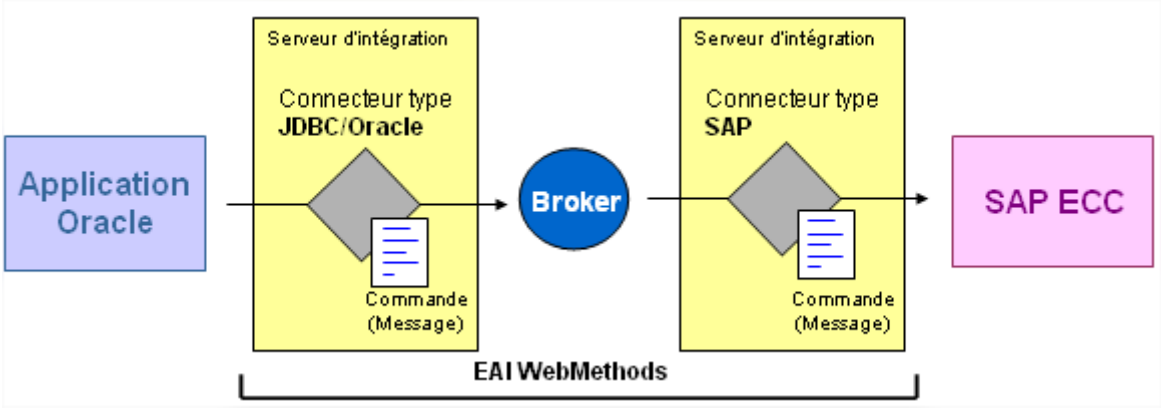


Illustration 13: Architecture d'une interface sur l'EAI WebMethods.

L'hétérogénéité des connecteurs constitue la force des EAI. Grâce à eux il est possible de faire communiquer facilement entre elles des applications techniquement très différentes. Parmi les connecteurs les plus repandus, le connecteur JDBC permet de s'interfacer avec la majorité des systèmes de gestion de bases de données. Il existe aussi évidemment un connecteur pour se connecter aux applications SAP, ou encore des connecteurs pour construire des fichiers plats ou au format XML, réaliser des échanges FTP, appeler des web services... Par ailleurs, dans le cas de systèmes très spécifiques ne pouvant être interfacés avec l'un des types de connecteurs existant, il est possible de réaliser soi-même un nouveau connecteur dans la mesure où l'on dispose d'une API - *Application Programming Interface* - ou de suffisamment d'informations techniques sur l'application concernée pour réaliser ce développement.
Les fonctionnalités B2B sont depuis quelques années incluses dans l'EAI WebMethods au travers des connecteurs web et XML, puis d'un outil de monitoring spécialisé nommé Trading Network.

En constituant un lieu de passage obligatoire pour la communication entre les systèmes, le composant « broker » de l'EAI WebMethods est un point unique de panne générale. En effet, en cas d'indisponibilité de ce dernier toutes les applications du système d'information se trouveraient isolées. Son usage et sa disponibilité doivent donc être attentivement maîtrisés.

Les messages échangés sur l'intergiciel ont un format entièrement défini au cours du développement des interfaces. Il est recommandé de les concevoir de manière la plus générique possible afin de garantir les possibilités de réutilisation, tel qu'elles seront décrites dans une prochaine section dédiée au modèle de publication et de souscription.
Dans l'industrie informatique, les formats des messages échangés sur les EAI sont communément appelés « documents canoniques ». Chez Givaudan ces mêmes documents sont appelés des BDM, acronyme de « Business Data Model » signifiant « Modèle de données métier ». Les spécifications des règles d'extraction et d'importation - *les règles de mapping avec les applications* - sont réalisées au travers d'un document standardisé au format Microsoft Excel qui sera présenté plus loin.

4.2.2 L'ETL DataStage

DataStage est un intergiciel ETL qui est la propriété d'IBM depuis 2005 suite au rachat de la société Ascential Software. Il est inclu dans la célèbre famille de produits WebSphere.

Spécialisé dans les traitements de type batch, DataStage dispose à l'instar des EAI d'une multitude de connecteurs afin de pouvoir se connecter à différents types d'applications. Le déclenchement des interfaces peut se faire en se connectant à l'outil de développement et d'exécution ou au travers d'un ordonnanceur externe; une solution plus adaptée pour les processus d'intégration contenant également des tâches externes à l'ETL.

Dans le but de simplifier la communication avec les équipes fonctionnelles, il fut décidé au cours du projet Outlook de documenter les spécifications des règles d'extraction et d'importation - *les mapping* - des interfaces DataStage de la même manière que pour les échanges de type EAI. Ainsi, les formats des données échangées sont aussi spécifiés dans des documents appelés BDM.

4.3 Le modèle de publication et de souscription

Le modèle de publication et de souscription est une stratégie d'intégration provenant du monde des EAI. Sur le projet Outlook, l'utilisation de cette stratégie est tant que possible étendue à la conception des interfaces sur l'ETL DataStage. Le but du concept est de réaliser des développements en demi-flux au travers de formats d'échange génériques afin de promouvoir l'indépendance, la réutilisabilité et la pérennité des développements.

Dans le modèle, on dit qu'une application « publie » des données lorsqu'elle transmet de l'information à l'intergiciel, les systèmes auxquels ce dernier envoie des données sont quant à eux dits souscripteurs. La publication et la souscription s'articule au niveau du format des messages et en aucun cas en fonction de l'identité de l'application qui publie les informations. Ainsi pour chaque type de message - *chaque BDM* - il est possible de disposer de plusieurs émetteurs et récepteurs. L'illustration ci-dessous représente un exemple d'application du modèle articulé autour du BDM « client ».

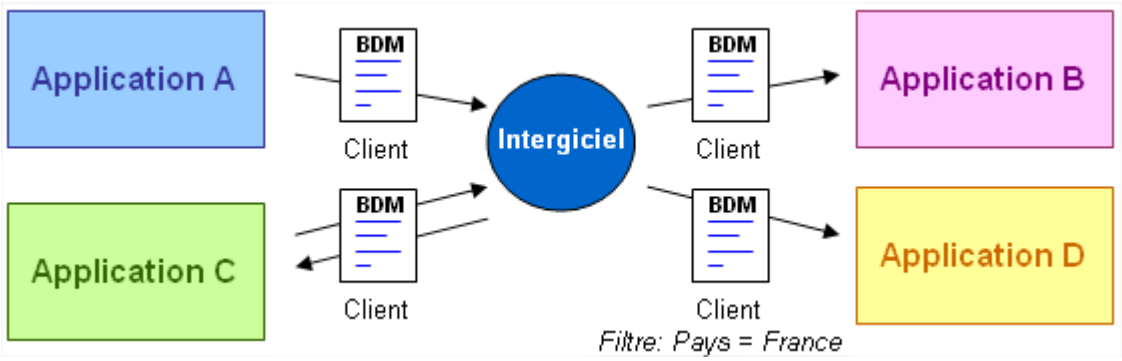


Illustration 14: Exemple d'application du modèle de publication et souscription.

Dans ce scénario, les applications A et C publient à l'intergiciel des informations au format du BDM client. B, C et D souscrivent à ce BDM avec la particularité pour l'application D de filtrer les informations reçues de telle manière à ne recevoir uniquement que les clients ayant la caractéristique d'être issus de France. Ce filtre est implémenté dans les développements réalisés sur l'intergiciel, il nécessite au BDM client de disposer de la zone « pays ». Ainsi, lorsque A ou B publient un message de type BDM client, ce dernier est reçu par les applications B et C, puis D dans le cas où le client est Français.

Chaque publication ou souscription par une application d'un format de message est réalisée dans l'intergiciel sur un espace de développement entièrement indépendant. Ainsi, les éventuels problèmes seront automatiquement isolés. Par ailleurs avec un tel modèle, remplacer une application du système informatique par une autre est relativement aisé car il s'agit uniquement de redévelopper les communications entre le nouveau système et l'intergiciel.
La plupart des nouvelles interfaces d'un système d'information sont premièrement développées dans un scénario où il existe uniquement un émetteur et un récepteur. Avec le temps et les évolutions du SI, de nouvelles applications viennent ensuite s'articuler autour de ces mêmes

formats de messages. Afin de garantir les facilités de réutilisation, il est recommandé de concevoir des BDM de format générique incluant le maximum de zones sur l'objet métier concerné. L'expérience démontre que le coût de la surcharge des BDM en information non utiles à la première utilisation est limité, voir inexistant. En revanche lorsque le temps sera venu de réutiliser ces BDM, la présence des champs additionnels évitera des développements et des phases de tests parfois risquées et coûteuses. Ainsi, travailler efficacement avec le modèle de publication et de souscription nécessite de se projeter dans l'avenir et de prévoir les futurs usages lors de la définition de nouveaux formats.

De par leur composant central qu'est le courtier de messages (broker), l'application de ce modèle dans le cadre d'un EAI est évidente. Pour reproduire ce concept avec un ETL il est nécessaire de rendre indépendantes les extractions (publications) et chargements (souscriptions) en enregistrant toujours les données de manière temporaire au milieu de l'exécution des flux d'intégration. La solution utilisée sur le projet Outlook est d'enregistrer les données dans des fichiers plats. Une fois créés, ces derniers sont ensuite utilisés pour alimenter une multitude de souscripteurs.

4.4 Intégration avec SAP

Les systèmes SAP disposent de plusieurs solutions techniques pour la mise en place des interfaces. Ce chapitre présente les solutions qui furent adoptées au cours du projet Outlook pour chaque type d'interface à mettre en place. Les communications B2B ne sont pas décrites ci-après car d'un point de vue SAP elles s'articulent exactement de la même manière que les interfaces de type asynchrone temps-réel.

4.4.1 Les communications synchrones

Tel qu'indiqué précédemment, le standard qui fut adopté pour les communications synchrones est l'usage de services web. La technique de développement des services web avec SAP repose sur la conception de modules de fonctions. Les modules de fonctions sont des morceaux de code développés en ABAP exécutant des traitements sur SAP. Ils peuvent disposer de plusieurs paramètres en entrée et en sortie.

Les appels externes à SAP sont habituellement réalisés sur des modules de fonctions à travers le protocole propriétaire RFC, acronyme de « Remote Function Call ». Pour procéder à la mise à disposition d'une fonction en tant que service web, il est nécessaire d'exécuter un outil de configuration pas à pas disponible dans les services de développement SAP. Une fois terminé, le module de fonction se trouve être accessible au travers du protocole SOAP - *Simple Object Access Protocol* - en tant que service web.
Les transactions SAP WSCONFIG et WSADMIN permettent ensuite de télécharger l'indispensable fichier WSDL - *Web Service Description Language* - à fournir aux développeurs du système exécutant l'appel afin qu'ils puissent exécuter sur leurs technologies les mécanismes de création de code automatique pour les services web.

Le développement sur SAP d'appels à des services web poursuit le procédé inverse. Les développeurs du système externe fournissent un fichier WSDL qui une fois chargé sur SAP à travers la transaction WSADMIN crée automatiquement un module de fonction comportant les paramètres d'entrée et de sortie spécifiés dans le fichier de description. Pour exécuter un appel du service, il suffit ensuite aux développeurs SAP d'exécuter le module de fonction avec les paramètres requis. Ce dernier fournira automatiquement en retour les données de réponse produites par le système externe, dans le cas où il est disponible.
L'illustration ci-dessous représente les appels synchrones entrants et sortants au travers des modules de fonctions ainsi que des protocoles de communication RFC et services web (SOAP).

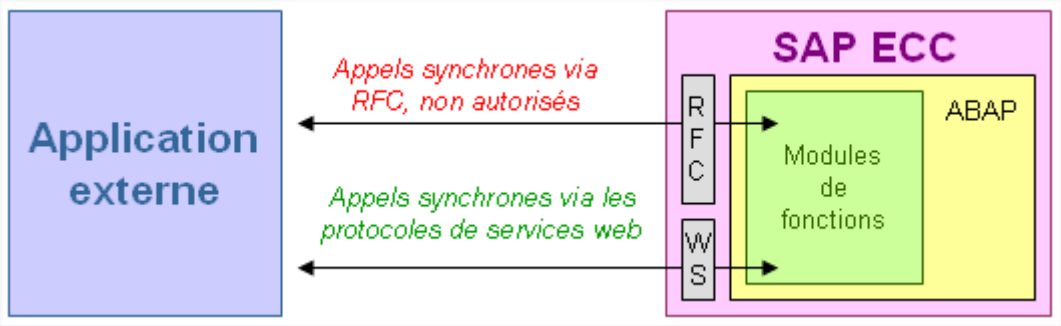


Illustration 15: Communications synchrones avec SAP,
Services web VS protocole RFC.

4.4.2 Les communications asynchrones temps-réel

SAP permet de réaliser des communications de type asynchrone temps-réel à travers une suite d'outils regroupés sous l'acronyme ALE pour « Application Link Enabling ». Les éléments de base pour la mise en place de ces interfaces sont les iDocs. Les paragraphes ci-après décrivent les iDocs ainsi que les principaux outils permettant de les administrer.

4.4.2.1 Les iDocs

Les iDocs, acronyme de « intermediate document » sont tels les BDM des formats de données destinés à réaliser des échanges. Ils sont caractérisés par trois principaux critères:

- iDoc type: un type d'iDoc fait référence à une structure de données composée de segments.
- Extension: pour chaque iDoc, disposer d'une extension est optionnel. Les extensions permettent d'étendre les structures de données des types d'iDoc avec des segments additionnels.
- Message type: chaque iDoc doit obligatoirement disposer d'un type de message. Il fait référence aux types de données contenues dans le message et parfois aussi à leur contexte.

Les types d'iDoc sont constitués de segments hiérarchisés en fonction de la structure des données à contenir. Tel un enregistrement d'une table, les segments comportent une liste de zones contenant au maximum une valeur pour chacune. Lors de la définition des types d'iDoc, les segments peuvent être intégrés comme étant itératifs afin de leur permettre de transférer des

informations multi-valuées, tel que les lignes d'une commande. L'illustration ci-dessous représente la structure d'un iDoc. Les segments E1EDK01 et E1EDKA1 sont disposés à la racine de l'iDoc ZINVOIC02A (extension de l'iDoc standard INVOIC02). Le premier segment comporte cinq sous segments, alors que le second n'en contient qu'un seul.

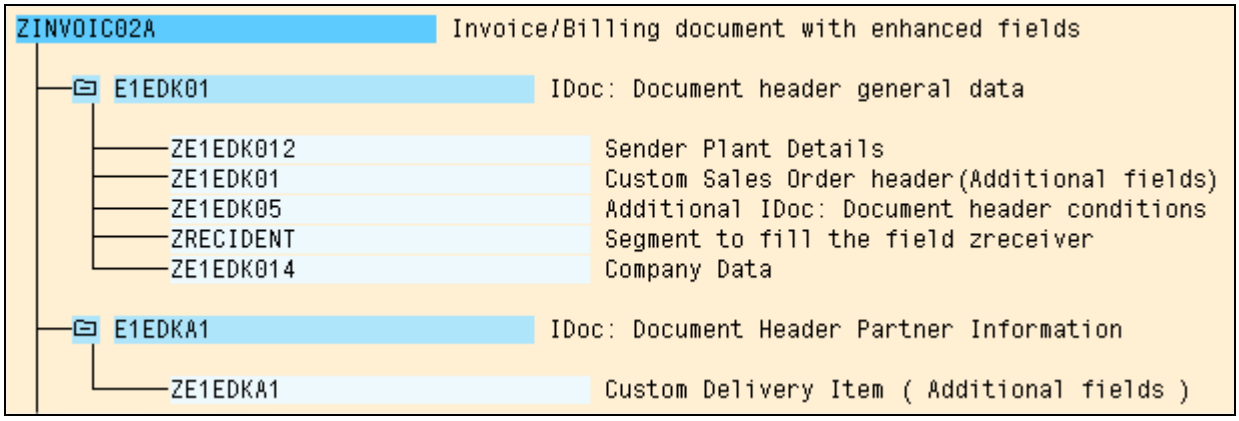


Illustration 16: Transaction WE32, structure d'un iDoc.

Que cela soit pour les interfaces entrantes ou pour les flux sortants, la communication entre l'EAI WebMethods et SAP repose sur des échanges d'iDocs. Pour une interface sortante, l'iDoc est construit par SAP, puis envoyé à WebMethods. Une fois le message reçu ce dernier applique des règles de mapping pour concevoir un BDM qu'il publie ensuite sur le broker. Les messages entrants appliquent le cycle inverse: une application publie un BDM sur WebMethods qui est ensuite souscrit par le connecteur SAP, ce dernier exécute des règles de mapping pour fabriquer un iDoc qui sera immédiatement transmis à SAP.

La transaction WE05 permet de visualiser les iDocs et leur état dans SAP. Chaque état est caractérisé par un numéro qui est différent selon que l'interface soit entrante ou sortante. Les principaux états sont listés ci-dessous:

- iDoc entrants: 64: en attente de traitement; 62: en cours de traitement; 53 terminé avec succès; 51: terminé avec erreur; 75: iDoc en file d'attente (lorsque le traitement en file est activé).
- iDocs sortants: 30: iDoc en attente de transfert; 03: iDoc transféré à la couche communication; 41: confirmation de réception par le système externe (EAI WebMethods).

Par défaut, le connecteur SAP de WebMethods ne gère pas la mise à jour des statuts. En conséquence, après leur transfert les iDocs transmis en sortie de SAP restent dans l'état « 03: iDoc transféré à la couche communication ». Pour pallier le manque de visibilité sur la bonne réception des messages par l'EAI, nous avons développé sur le connecteur SAP de WebMethods un mécanisme de confirmation de réception. Ainsi au bout de cinq minutes, les iDocs reçus par l'EAI sont dorénavant automatiquement disposés dans l'état « 41: confirmation de réception ».

4.4.2.2 Configuration ALE

La configuration ALE permet de renseigner dans le système SAP les paramètres nécessaires aux transferts et traitements des iDocs. Cette tâche repose sur la définition des profils de partenaires - *partner profiles* - et du modèle de distribution - *distribution model*.

Les profils sont configurés au travers de la transaction WE20. Pour chaque type de message en sortie de SAP, il est nécessaire de renseigner les informations suivantes:

- Nom du type l'iDoc et nom de l'extension (si applicable).
- Transfert immédiat à activer ou non. Si le transfert immédiat est actif, alors à sa création l'iDoc sera automatiquement transmis à la couche communication. Dans le cas inverse, le message restera en attente dans SAP et sera transféré uniquement à l'exécution d'un programme de transfert.
- Taille de paquet: valeur numérique indiquant le nombre maximum d'iDocs pouvant être transmis au cours d'une même communication réseau. Applicable uniquement si le transfert immédiat est désactivé.
- Port de réception: information technique identifiant la connexion avec le système externe.

Les messages en entrée de SAP requièrent quant à eux d'avoir les informations suivantes renseignées dans leur profil:

- Code du processus: identifiant ABAP permettant de retrouver le programme à utiliser pour traiter le message.
- Exécution immédiate à activer ou non. Si activé, alors à la réception l'iDoc sera automatiquement traité. Dans le cas inverse il patientera l'exécution d'un programme de déclenchement des traitements.

Activer les options de traitement et de transfert immédiat permet de réduire la durée de synchronisation des systèmes (la latence). Néanmoins l'impact d'une telle configuration sur les performances du progiciel SAP nécessite d'être pris en considération.

Accessible par la transaction BD64, le modèle de distribution permet quant à lui d'assigner des types de messages aux profils de partenaires. Son usage est nécessaire que pour certaines des interfaces sortantes du système. Il permet par ailleurs de spécifier des filtres afin de générer des iDocs uniquement quand certaines conditions sur les données sont rencontrées.

4.4.2.3 Pointeurs de modification

Les pointeurs de modification permettent de traquer les changements réalisés par les utilisateurs ou les programmes sur les objets métiers SAP. Ils sont ainsi une solution efficace pour construire les mécanismes de déclenchement des interfaces.

Lorsqu'un objet métier SAP traqué par le mécanisme des pointeurs de modification est changé, la configuration des déclencheurs est lue, puis pour chaque zone de l'objet ayant subit une modification, des enregistrements sont ajoutés dans la table SAP BDCP2. L'illustration ci-

dessous représente des pointeurs notifiés dans la table BDCP2 pour l'interface ayant pour type de message ZGLHUINFO.

Messg.Type	Chng.pntr	P	Table	Table Keys Long	Field Name	Creation time	Object value	Change ID
ZGLHUINFO	519741398		VEKP	1000021250220	VPOBJKEY	201009200945	0021250220	U
ZGLHUINFO	519741399		VEKP	1000021250220	WERKS		0021250220	U
ZGLHUINFO	519740745	X	HUSSTAT	100HU00212502	KEY	201009200935	0021250213	I
ZGLHUINFO	519740746	X	HUSSTAT	100HU00212502	KEY		0021250213	I
ZGLHUINFO	519740747	X	HUSSTAT	100HU00212502	KEY		0021250213	I
ZGLHUINFO	519740748	X	HUSSTAT	100HU00212502	KEY		0021250213	I

Illustration 17: Pointeurs de modification SAP écrits pour le type de message ZGLHUINFO.

Lorsque la zone P, troisième à partir de la droite, contient la valeur X alors le déclencheur de l'interface concernée a déjà traité le pointeur. En revanche, si cette même colonne est vide, alors le pointeur sera traité à la prochaine exécution du programme d'analyse. La colonne change ID indique le type de changement ayant eu lieu: I pour une création, U pour une modification et D pour un effacement des données. La table BDCP2 contient par ailleurs d'autres zones non représentées ci-dessus qui permettent par exemple d'identifier la personne ayant exécuté le changement ou d'obtenir des détails complémentaires sur l'événement concerné.

La configuration des déclencheurs se fait au moyen de la transaction BD60. Pour chaque type de message, des tables et des champs du système SAP sont assignés. Lorsque ces zones seront changées, alors un nouveau pointeur sera automatiquement généré. Malheureusement, il n'est pas possible de configurer le mécanisme des pointeurs sur l'ensemble des objets et tables de SAP. Il est en conséquence parfois nécessaire de concevoir des mécanismes de déclenchement alternatifs.

Pour déclencher le programme d'analyse des pointeurs il faut utiliser la transaction BD21 et renseigner le type de message dans l'unique zone de saisie de l'écran de sélection. Une fois le programme exécuté, alors tous les pointeurs non traités correspondant au type de message entré seront marqués et les iDocs auront été générés si les données sont conformes aux éventuels filtres définis dans le modèle de distribution.

4.4.2.4 Exécution

La mise en place de communications avec SAP au moyen d'échange d'iDocs nécessite de configurer l'exécution automatique en arrière plan de multiples programmes. La majorité des programmes SAP disposent d'un écran de sélection permettant de spécifier des paramètres d'entrée. Les valeurs renseignées dans ces paramètres peuvent être sauvegardées sous le nom de « variantes » afin de pouvoir être rechargées ultérieurement ou de servir profil à l'ordonnancement d'un programme en arrière plan.

Lors de la définition des automatismes, il fut décidé de proposer pour chaque interface trois fréquences de traitement au choix: 1, 5 et 15 minutes. Dans la pratique, une fréquence de 5

minutes s'avère convenir à la majorité des cas, néanmoins certaines applications tierces exécutant des mouvements de stock nécessitent parfois d'être synchronisées en moins d'une minute.

Les principaux programmes de traitement des interfaces sont présentés ci-dessous:

- RBDAPP01: programme exécutant tous les iDocs reçus par le système SAP (état 64). A l'issue de son exécution, les iDocs seront disposés dans l'état 53 (succès) en cas de fonctionnement normal ou 51 (échec) en cas d'erreur.
- RBDMANI2: programme exécutant tous les iDocs reçus par le système SAP et ayant échoué lors de leur traitement avec RBDAPP01 (état 51). Si la nouvelle exécution du traitement fonctionne, alors l'iDoc sera disposé dans l'état 53 (succès).
- RSEOUT00: programme déclenchant le transfert vers la couche communication de tous les iDocs en sortie produits par SAP (état 30). Une fois exécuté, les iDocs transférés seront disposés l'état 03 (transféré).
- RSEINBQUEUE: programme déclenchant le traitement des iDocs reçus par le système SAP dans une file d'attente (état 75). Ce programme exécute chaque iDoc en séquence puis s'arrête lorsque la file d'attente est vide ou à la première erreur de traitement afin de ne pas briser la chaîne d'exécution. Tel que pour le programme RBDAPP01, les iDocs sont ensuite disposés dans les états 51 (échec) ou 53 (succès).
- RBDMIDOC: Similaire à la transaction BD21, ce programme lit les pointeurs de modification et génère les iDocs correspondants après avoir consulté les filtres définis dans le modèle de distribution.
- RSARFCEX: En cas de problème réseau ou d'indisponibilité de l'EAI connecté, ce programme permet de re-déclencher les transferts une fois que le système distant est de nouveau connecté.

Les principes de fonctionnement de ces programmes standard ne sont pas toujours adaptés à l'ensemble des traitements. Il est donc parfois nécessaire de développer des programmes alternatifs incluant des logiques spécifiques. Par exemple lors de leur exécution, RBDAPP01 et RSEOUT00 sélectionnent tous les iDocs présents dans le système qui correspondent à leurs critères de sélection, puis les regroupent par type de message. Les traitements sont ensuite réalisés en procédant groupe par groupe. Un tel procédé à pour effet de casser l'ordre naturel de traitement des messages et peut donc mener à des résultats inconsistants.

Bien que les erreurs de traitements soient censées disposer les iDocs dans un état particulier, il arrive que le programme soit entièrement annulé en raison de problèmes de programmation ou d'inconsistances importantes dans le système. Fréquentes pendant les phases de stabilisation, de telles erreurs sont appelées « dump ». Pour éviter les impacts de tels problèmes il est préférable de ne pas regrouper tous les traitements dans une unique variante. Le choix qui fut entériné sur le projet Outlook est donc de disposer de programmes ordonnancés dédiés pour chaque univers métier. Ainsi, les domaines fonctionnels finance, fabrication et distribution disposent pour chacun d'un programme ordonnancé dédié aux traitements de leurs interfaces.

4.4.3 Les communications batch

Les développements d'interfaces batch au moyen des outils ETL sont fréquemment basés sur des accès direct aux tables des bases de données des applications. Or, pour des raisons liées à la sécurité et aux restrictions imposées par les contrats de support, les accès à la base de données de SAP au travers d'outils externes sont entièrement prohibés tant en écriture qu'en lecture.

Conscient de cette limitation, nous avons premièrement testé une extension de DataStage permettant de se connecter à SAP au travers de la couche RFC. Néanmoins, le coût et la complexité de cet outil mis en relation avec les éventuels problèmes que nous aurions eu pour trouver des ressources capables d'y réaliser des développements nous a amené à rechercher d'autres alternatives. C'est ainsi que nous avons imaginé une solution technologiquement simple et robuste fonctionnant au travers de fichiers plats. Tel qu'illustré ci-après, les données sont extraites de SAP par des programmes ABAP exécutant des requêtes à la base de données, puis transférant leurs résultats vers des fichiers plats. Ces fichiers sont ensuite déplacés sur le serveur de l'ETL, puis les processus de l'intergiciel sont déclenchés pour lire les données et procéder aux étapes propres à l'intégration avec le système tiers. Parfois, une dernière action qui consiste à déclencher un programme de traitement sur l'application réceptrice est de plus requise. Toutes ces activités sont coordonnées grâce à ControlM, l'ordonnanceur de tâches de Givaudan.

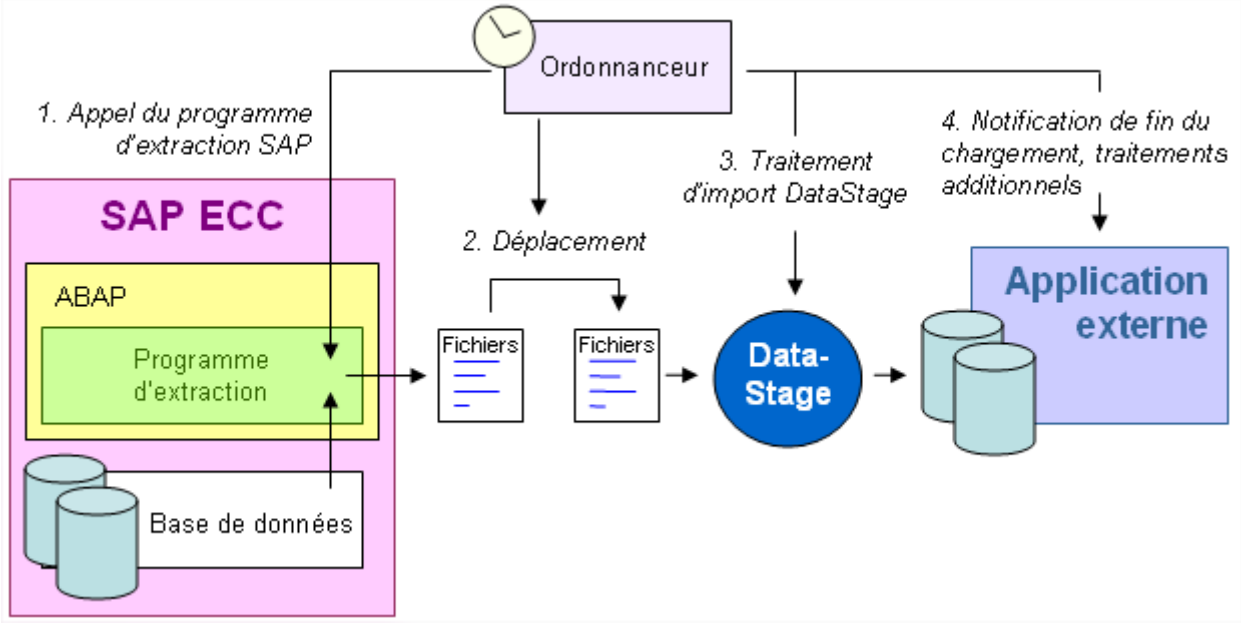


Illustration 18: Communications batch avec le progiciel SAP.

Les fichiers extraits de SAP peuvent être utilisés pour déclencher plusieurs processus DataStage alimentant pour chacun une application différente. C'est ainsi que le modèle de publication et souscription fut adapté aux communications batch.
Une telle architecture dispose de plus d'un avantage conséquent au niveau de l'organisation du projet. Peu familiers avec SAP, les développeurs DataStage peuvent travailler de manière

autonome avec des fichiers simplement structurés. La création des programmes ABAP est quant à elle donnée au centre de développement SAP, dans lequel travaillent pour la plupart des personnes ne disposant pas d'expertise technique dans le domaine des interfaces.

4.5 Spécifications

Les nouveaux besoins en développement sont spécifiés par les analystes système au travers d'un document Microsoft Word appelé « spécification fonctionnelle ». Dans le cadre des interfaces, ce document est complété d'un fichier au format Microsoft Excel décrivant la structure des données du BDM à créer.

Cette section présente les fichiers de description des BDM ainsi que le modèle de spécification des traitements ordonnancés.

4.5.1 Règles d'extraction et d'importation

Les structures des BDM ainsi que les règles d'extraction et d'importation de chaque application publiant ou souscrivant au message électronique sont spécifiées au travers d'un fichier standardisé. Créés avec Microsoft Excel, ces documents aussi appelés « fichiers de mapping » sont constitués de plusieurs feuilles de travail (onglets), chacune étant décrite ci-après.

- Onglet « Informations » :

Informations générales sur le BDM: nom, description générale de l'objet ou l'événement qu'il est censé transférer, nom du créateur, date et historique des modifications, listing des applications publiant ou souscrivant au document.

- Onglet « Structure » :

L'onglet structure décrit le format du BDM de manière hiérarchique. Chaque champ est caractérisé par un nom, un type (nombre, date ou texte) et une taille. Situé tout à droite, une colonne du formulaire est dédiée à la description du champ de la ligne concernée. Par ailleurs chaque zone et chaque niveau hiérarchique peut être défini comme itératif afin de pouvoir contenir plusieurs valeurs. La création et la mise à jour de l'onglet structure du BDM est la responsabilité des équipes fonctionnelles. L'illustration ci-dessous représente de manière simplifiée la définition de la structure d'un BDM composé de 6 zones.

Structure	Zones	Iteratif	Type	Size	Description
Commande					
	Numéro		Nombre	10	Numéro de commande
	CodeClient		Nombre	8	Code du client
	Date		Date		Date de création
Ligne		Y			
	CodeProduit		Chaine	8	Code produit
	Quantité		Nombre	6	Quantité en KG
	Prix		Nombre	8,2	Prix en Euros

Illustration 19: Exemple simplifié d'onglet structure dans un fichier de mapping BDM.

- Onglet « *Application* pub » :

Chaque application publiant le BDM concerné dispose d'un onglet « *Application* pub » dans lequel les règles de mapping de publication sont spécifiées. Basé sur la structure du BDM, le document spécifie pour chaque champ une relation vers une zone d'un iDoc, ou d'une table selon que l'application fonctionne sur SAP ou sur un système auquel l'intergiciel peut directement accéder à la base de données. Lorsque ce système est SAP alors le traitement est déclenché par la réception d'un iDoc tandis que lorsque l'intergiciel extrait les données à partir d'une base de données un mécanisme de notification est mis en place au travers de sélections fréquentes dans une table créée spécifiquement. Illustré ci-après, un exemple d'onglet de publication avec une application SAP.

					iDoc EXAMPLE01, Message type EXAMP	
Structure	Zones	Iteratif	Type	Taille	Segment	Champ / Logique
Commande						
	Numéro		Nombre	10	E1VBAP	
	CodeClient		Nombre	8	E1VBAP	
	Date		Date		E1VBAP/E2BRKV	Format YYYYMMDD
Ligne		Y			*Une ligne pour chaque itération de E1VBLN*	
	CodeProduit		Chaine	8	E1VBLN	MATNR
	Quantité		Nombre	6	E1VBLN	QUANT
	Prix		Nombre	8,2	E1VBLN	PRICE

Illustration 20: Exemple de mapping de publication à partir d'un iDoc.

- Onglet « *Application* sub » :

A l'inverse de « *Application* sub », l'objectif de cet onglet est de contenir les mappings de souscription. Les transferts se font au travers d'iDoc pour les systèmes SAP, ou généralement via des accès aux tables des applications pour les autres systèmes, tel que dans l'exemple suivant.

					Application MONAPPLI	
Structure	Zones	Iteratif	Type	Taille	Table	Champ / Logique
Commande						
	Numéro		Nombre	10	CO_ENTETE	NUMERO
	CodeClient		Nombre	8	CO_ENTETE	CODECLI
	Date		Date		CO_ENTETE	CODATE
Ligne		Y			*Insertion dans CO_LIGNE pour chaque ligne*	
	CodeProduit		Chaine	8	CO_LIGNE	PRODUIT
	Quantité		Nombre	6	CO_LIGNE	QUANT
	Prix		Nombre	8,2	CO_LIGNE	PRIX

Illustration 21: Exemple de mapping de souscription utilisant des tables.

Lorsque les BDM disposent de multiples publieurs et souscripteurs, alors les onglets « *Application* pub » et « *Application* sub » sont dupliqués pour chaque application. Il est par ailleurs possible qu'une même application publie et souscrit à la fois au même BDM. Elle disposera dans ce cas de deux onglets dans le fichier de mapping.

4.5.2 Traitements ordonnancés

Les traitements ordonnancés sont uniquement utilisés dans le cas des interfaces de type batch. Les communications au travers d'iDocs et de l'EAI requièrent aussi des mécanismes de déclenchements mais ces derniers sont exécutés à des intervalles très réguliers sans être interdépendants.
Le document de spécification des traitements ordonnancés est un fichier au format Microsoft Excel dans lequel les tâches et les conditions de gestion des exceptions sont précisées dans la première feuille de travail (onglet). Les suivantes, spécifiques à chaque environnement, indiquent les paramètres à utiliser pour les environnements concernés (développement, qualité, test, production...).

Chaque tâche est spécifiée de la manière suivante:

- Condition de déclenchement: A quel moment déclencher la tâche. Les premières tâches des traitements sont généralement déclenchées à heure fixes tandis que les autres s'exécutent à la fin de la tâche précédente.
- Durée estimée en minutes
- Application ou outil exécutant le traitement: SAP, DataStage ...
- Nom du programme à appeler
- Paramètres : Variante dans le cas de SAP. Les paramètres renseignés dans cette zone peuvent être en relation avec des valeurs définies dans les onglets dédiés aux différents environnements.

4.6 Organisation projet

La taille et la complexité du projet Outlook requiert de disposer de procédures strictes et d'outils facilitant le suivi du travail. Cette section présente le paysage applicatif qui fut construit pour le projet, l'outil de gestion de la qualité et le processus de gestion des transports des développements.

4.6.1 Le paysage applicatif

Le paysage applicatif du projet Outlook est composé de cinq environnements principaux.

Au plus bas niveau, l'environnement sand-box (traduction: bac à sable) est un système SAP utilisé pour réaliser et tester des prototypes. Les développements créés sur ce système ne peuvent pas être transférés sur les autres environnements, ils devront être copiés manuellement. Par ailleurs, sand-box est un système isolé dans le sens où il n'est pas connecté à d'autres applications au travers d'interfaces.

Premier de la chaîne de transport, l'environnement de développement est celui où les développements et les tests unitaires sont réalisés. Une fois terminé, le code est transporté vers le système Qualité, puis le système de business simulation (traduction: simulation métier) où les tests sont respectivement exécutés par les équipes fonctionnelles et les utilisateurs. Chacun de ces

trois environnements comporte un environnement WebMethods dédié. L'intergiciel DataStage ne dispose quant à lui que de deux instances non-productives car le système de test est en mesure d'héberger et d'exécuter les traitements de deux environnements SAP distincts. Quant aux applications tierces, elles disposent pour la plupart d'environnements dédiés pour chacun des systèmes SAP. Néanmoins, il est parfois nécessaire de partager les systèmes en fonction des cycles de test du projet.

Au bout de la chaîne, l'environnement de production dispose quand à lui logiquement d'instances dédiées et entièrement isolées des systèmes non-productifs.
L'illustration ci-dessous représente le paysage applicatif du projet Outlook. Dans l'organisation Givaudan, les environnements sont généralement nommés par leurs acronymes: SBX, DEV, QAS, BUS, PRD.

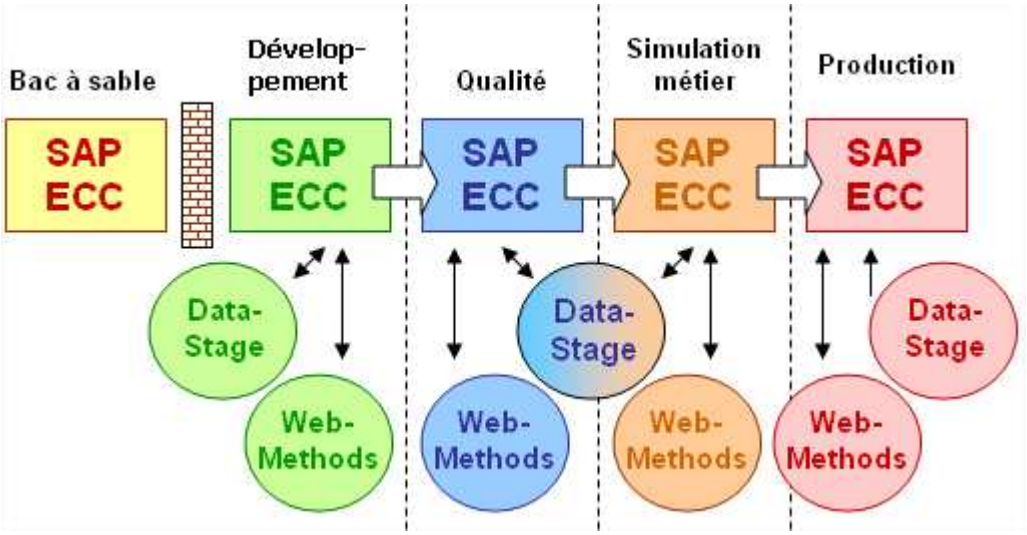


Illustration 22: Le paysage applicatif du projet Outlook.

Il existe par ailleurs d'autres environnements SAP pour répondre à des besoins temporaires du projet tels que la réalisation de tests de migration ou la simulation de nouvelles configurations. Ces systèmes ne font pas parti de la chaîne de transport du projet et ne sont pas connectés à des applications tierces.

4.6.2 L'outil de gestion de la qualité

La gestion de la qualité sur le projet Outlook est assurée au travers de l'outil Mercury Quality Center édité par la société HP. Sous forme d'application web, Mercury permet de suivre l'exécution des tests ainsi que de gérer le suivi des problèmes.

Sur le projet, l'usage de la liste de problèmes référencés sur Mercury fut étendu à la gestion des demandes de changement au travers d'une zone « classification ». Chaque entrée est donc caractérisée par un numéro d'identifiant (defect number), un type (defect ou change request), le nom d'utilisateur de la personne en charge du prochain traitement, un statut de la demande

incluant la position du développement correspondant sur le paysage applicatif, un numéro de version ainsi que d'une multitude d'autres zones décrivant le contenu et permettant au projet de suivre l'évolution de la demande. L'illustration ci-après représente un sous ensemble de la liste des problèmes et des changements enregistrés dans Mercury.

Defect ID	Classification	Version	Status	Assigned To	Priority	Detected By	Summary
42179	Change Request	4.71	29 Successfully Tested in QAS	robialb	1 Critical	ryznerj	NOAM_FL_D
26185	Change Request	4.0	50 Defect closed	pattabij	2 High	pattabij	PRD_ALL sit
26103	Defect	3.0	50 Defect closed	kolki	1 Critical	pattabij	Packing infor
22487	Defect	2.51	50 Defect closed	gassmani	1 Critical	striging	PRD_VE: Cha
21680	Defect	2.5	08 Rejected	garciad1	3 Medium	garciad1	PRD_DO Wrc

Illustration 23: Problèmes et changements listés dans l'outil Mercury Quality Center.

4.6.3 La gestion des transports

Afin de garantir la stabilité du système, les transports des développements se doivent de suivre un processus strict incluant de la coordination pour l'exécution des tâches sur les différents systèmes impactés: SAP, applications tierces, intergiciels EAI et ETL.

Lors de chaque développement sur SAP, au moment de l'enregistrement du code le système réclame au développeur de fournir une clef appelée requête de transport. L'ensemble du nouveau code et des objets impactés se trouve ainsi automatiquement attaché à la requête qui constituera la référence unique du transport sur tout le paysage applicatif. Si plusieurs développeurs travaillent en sauvegardant leurs développements sur la même requête de transport, alors des tâches seront créées pour chacun d'eux sur cette dernière. Une fois que la requête est transférée vers un autre environnement, alors elle est automatiquement fermée et il faudra en ouvrir une nouvelle pour procéder à de nouveaux changements sur le même programme. L'illustration ci-dessous représente une requête de transport SAP assignée à un groupe nommé « MA4_73 ». Tous les développements de ce groupe seront transportés à la même date dans les environnements de simulation métier et production.

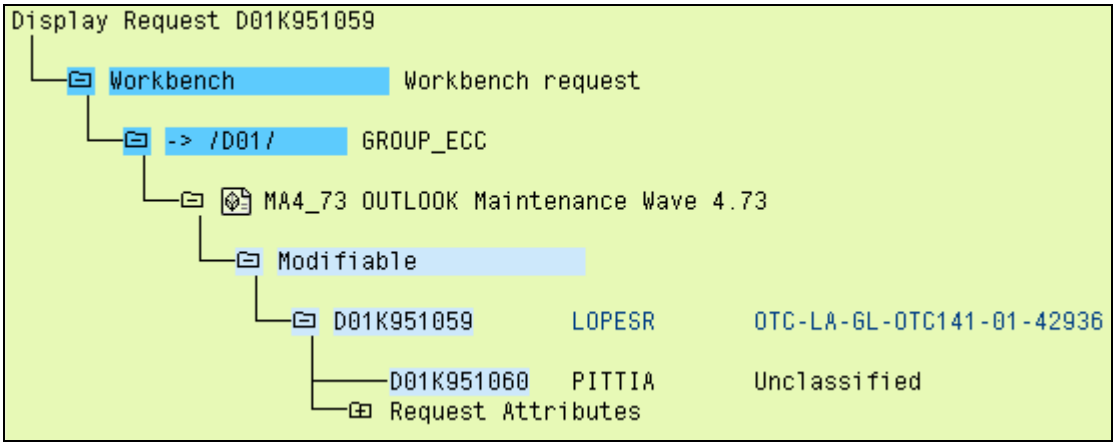


Illustration 24: Requête de transport SAP assignée au projet MA4_73.

Afin d'assigner chaque développement à une entrée Mercury de type erreur ou demande de changement, un outil de création de transport fut créé ainsi qu'une interface répliquant vers SAP les éléments de la liste des problèmes de l'outil de gestion de la qualité. Ainsi, il n'est plus possible de créer des requêtes de transport SAP sans les assigner à une erreur ou à une demande de changement suivie sur le projet. Grâce à ce lien, tous les transports sont gérés directement sur l'outil Mercury.

Les intergiciels WebMethods et DataStage ne disposent pas de mécanismes de clefs de transport similaire à ce qu'il existe sur SAP. Pour pallier ce problème important dans un contexte constitué de nombreux développements, un outil spécifique appelé TrackDev fut conçu afin de suivre les transports relatifs aux intergiciels. Tel que pour SAP, cet outil est lui aussi connecté via une interface à la liste des problèmes référencés sur Mercury. Les développements sur les applications tierces ne disposent quant à eux d'aucun service de suivi des transports. Réalisé manuellement, ce suivi requiert ainsi beaucoup de rigueur.

La gestion de l'ensemble des transports du projet est assurée par une équipe dédiée nommée « Outlook release managers ». Cette équipe définit un calendrier de transport ainsi que des numéros de version. Chaque transport se doit de suivre le calendrier, mis à part ceux dont la classification dans Mercury est spécifiée comme urgente. Pour éviter les abus, un processus de validation des transports urgents a été mis en place sur le projet. L'illustration ci-dessous représente un exemple du calendrier de transport Outlook incluant les versions allant de 4.6 à 4.73.

Version	Content	Transport Project	Date of transport to QAS	Date of transport to BUS	Date of transport to PRD
MAINTENANCE Waves					
4.71	Maintenance	MA4_71	From Mo 6 Sept	Mo 4 Oct 12:00pm CET	Su 17 Oct
4.72	Maintenance	MA4_72	From Mo 18 Oct	Mo 1 Nov 12:00pm CET	Su 21 nov
4.73	Maintenance	MA4_73	From Mo 22 Nov	Mo 6 Dec 12:00pm CET	Su 19 Dec
ROLL OUT Waves					
4.6	NOAM wave 2	PRJ_NA2	From Mo 24 May	Mo 12 Jul 12:00pm CET	Sa 04 Sept
4.7	Rest of EMEA	PRJ_EMEA	From Mo 24 May	Mo 12 Jul 12:00pm CET	Sa 04 Sept

Illustration 25: Le calendrier de transport des versions 4.6 à 4.73 du projet Outlook.

5 Cas 1: Intégration des clients et de leur hiérarchie avec l'application CDM

Pour tout progiciel de gestion intégré installé dans une entreprise œuvrant dans le domaine de l'industrie, les données sur les clients constituent des éléments essentiels à l'exécution de la majorité des processus métier. A travers le module *Sales and Distribution* (SD) SAP dispose d'outils complets pour gérer les informations sur les clients ainsi que la hiérarchie de ces derniers. Hiérarchiser les clients en fonction de leur structure en divisions et filiales permet de gérer des politiques de prix ou des paramètres de configuration fonctionnelle au niveau le plus fin ou le plus général selon les besoins du métier. L'illustration ci-après représente un exemple de hiérarchie pour le groupe français PSA.

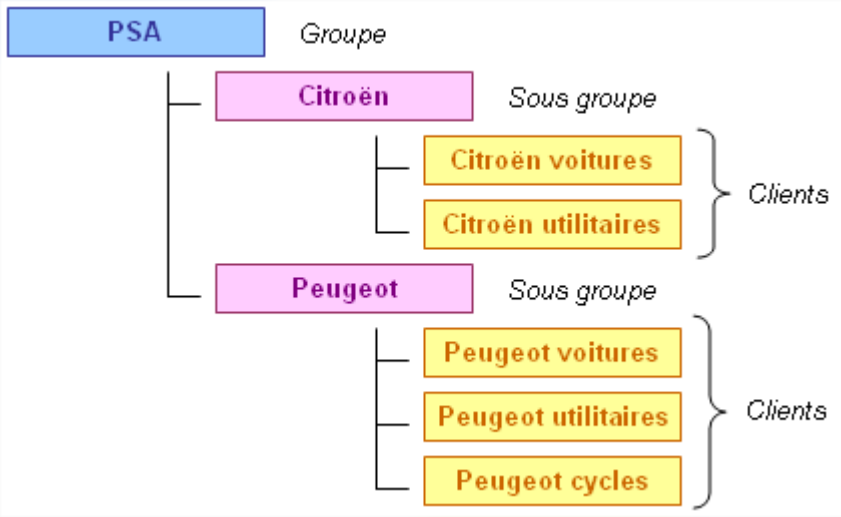


Illustration 26: Exemple de hiérarchie de clients pour le groupe PSA.

Grâce à cette modélisation, il est possible pour une entreprise d'équipement dans le domaine de l'automobile de disposer de politiques de prix qui seront définies soit pour l'ensemble du groupe PSA, soit pour un sous groupe ou un client en particulier.

Lors des analyses fonctionnelles, il fut décidé d'adapter à SAP le processus utilisé avec BPCS, l'ancien PGI de Givaudan. Ainsi, les nouveaux clients doivent être créés directement dans le système SAP tandis que la hiérarchie de ces derniers est maintenue dans une application tierce appelée CDM, acronyme de « Customer Data Management ». Il est donc nécessaire de mettre en place deux interfaces. La première, de SAP vers CDM aura pour objectif d'envoyer à ce second tous les nouveaux clients ainsi que les mises à jour sur leurs données. La seconde, de CDM vers SAP chargera la hiérarchie des clients et la synchronisera à chaque changement.

Au cours des prochains paragraphes une étude de conception analysera en détail ces deux interfaces. La phase de migration des données ainsi que le fonctionnement du processus de gestion des clients en condition d'exploitation seront aussi abordés.

5.1 Analyse fonctionnelle des besoins

Chaque client SAP dispose d'un type de compte indiquant l'usage fait de l'objet client dans les processus métier. Les principaux types de comptes sont les suivant:

- Client de vente (sold-to customer): ce type de client peut être utilisé dans les processus de vente et de distribution.
- Client de livraison (ship-to customer): les clients de livraison sont toujours rattachés à un ou plusieurs clients de vente, ils représentent les destinataires auxquels la marchandise sera livrée.
- Client de type groupe (group customer): noeud groupe de la hiérarchie des clients
- Client de type sous-groupe (sub-group customer): noeud sous-groupe de la hiérarchie des clients

A la création de chaque commande dans le système, il est nécessaire d'indiquer un client de vente qui sera unique pour la commande. Chaque ligne pourra en revanche se voir attribuer un client de livraison différent à condition que ces derniers soient tous rattachés au client de vente. Ceci rend possible la réception de la marchandise dans diverses usines pour une commande passée par un centre d'achat centralisé. Si le client de livraison au niveau de la ligne n'est pas précisé, alors c'est le client de vente qui recevra la marchandise.

Seuls les clients de vente doivent être transférés de SAP vers CDM. Les clients de type groupe et sous groupe sont créés directement dans CDM et devront être répliqués dans SAP au travers de l'interface de transfert des hiérarchies.
Avec près de vingt applications connectées, CDM un composant important du système informatique de Givaudan. En conséquence, en plus de satisfaire les besoins du projet Outlook, les données sur les clients qui seront transférées depuis SAP devront aussi permettre de garantir le fonctionnement des applications déjà connectées à CDM.

SAP étant un système global, chaque client créé dispose d'un identifiant unique. Un même client exerçant du commerce sur deux sites différents disposera d'une extension de vente pour chacun dans lesquelles les données spécifiques seront précisées. Lors de leur création dans le système les clients ne disposent pas d'extension de vente, c'est plus tard lorsque le moment de la première demande commerciale est venu que la première extension est créée. Si par la suite le client fait du commerce avec d'autres sites Givaudan, alors une extension de vente sera ouverte pour chacun. L'illustration ci-dessous représente la structure d'un objet client dans SAP.

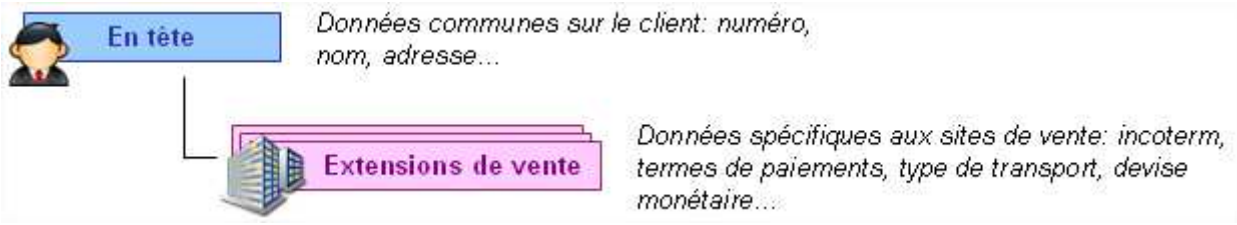


Illustration 27: Structure des données d'un client sur le PGI SAP.

La structure hiérarchique requise dans SAP est composée de trois niveaux: groupe, sous groupe et client de vente. Néanmoins CDM dispose d'un palier additionnel nommé « client unique » qui est situé entre les niveaux sous groupe et client de vente. Dans un environnement composé de multiples instances de BPCS, l'intérêt de ce niveau supplémentaire était de regrouper sous le même identifiant un unique client disposant de numéros différents dans chacun des sites. Avec SAP ce niveau n'est plus requis car comme le système est constitué d'une unique instance, chaque client dispose d'un identifiant global. En conséquence comme il est désormais inutile, le niveau unique ne doit pas être modélisé dans le nouveau progiciel.

Le dernier niveau de la hiérarchie doit être dupliqué pour chaque site de vente. Ainsi, un client disposant d'une extension sur deux sites devra avoir deux lignes à l'intérieur du même sous groupe. Si une troisième extension est ouverte sur ce même client, alors l'interface d'extraction sera déclenchée à nouveau puis la hiérarchie mise à jour automatiquement avec une troisième ligne dans le sous groupe. L'illustration ci-dessous représente un exemple de hiérarchie.

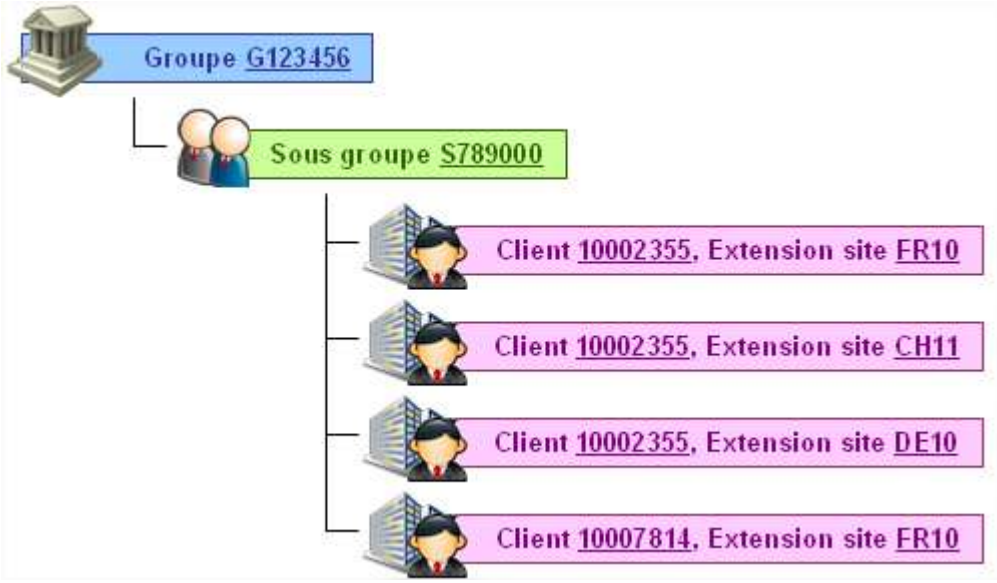


Illustration 28: Exemple de structure hiérarchique avec les clients 10002355 et 10007814 assignés au même sous groupe.

A leur création dans SAP, les clients ne disposent pas de hiérarchie dans CDM. Pour permettre aux administrateurs des données clients d'exécuter cette classification, une hiérarchie fictive doit être créée dans CDM. Chaque nouveau client sera enregistré dans cette hiérarchie fictive avant d'être définitivement assigné à un sous groupe correspondant. Composé d'un groupe et un sous groupe fictif, cette hiérarchie ne doit pas être répliquée dans SAP.

Les changements des clients dans SAP ou de la hiérarchie dans CDM doivent être répliqués en quelques minutes. Par ailleurs CDM classifie aussi des clients provenant d'autres systèmes et n'existant pas dans SAP. L'interface de transfert des hiérarchies ne doit donc pas être perturbée par les clients inconnus du système SAP ECC.

5.2 Architecture de la solution

La faible latence requise pour les deux interfaces requiert l'utilisation d'un intergiciel de type EAI. La communication avec SAP doit donc se faire au travers d'échanges d'iDocs via l'outil WebMethods. Tandis qu'un nouveau BDM nommé « CustomerMaster » devra être conçu pour l'extraction des clients de SAP, la hiérarchie des clients dispose déjà d'une interface en sortie de CDM qu'il sera possible de réutiliser grâce au BDM existant « CDMReplicator ». Néanmoins SAP et CDM étant deux systèmes de générations différentes, les codes utilisés par ces derniers pour identifier les sites de vente et lieux de production ne sont pas les mêmes. Pour combler ce problème et permettre aux applications de communiquer dans un même langage, une base de données de translation fut créée pour l'ensemble des interfaces du projet Outlook. Uniquement à l'usage des intergiciels WebMethods et DataStage, la base de données de translation sera ici utilisée pour l'interface « CustomerMaster ». La translation des données du BDM « CDMReplicator » sera quant à elle exécutée dans SAP afin de ne pas perturber les systèmes déjà construits autour de cette extraction. Le schéma ci-après représente la structure de ces deux interfaces entre SAP et CDM.

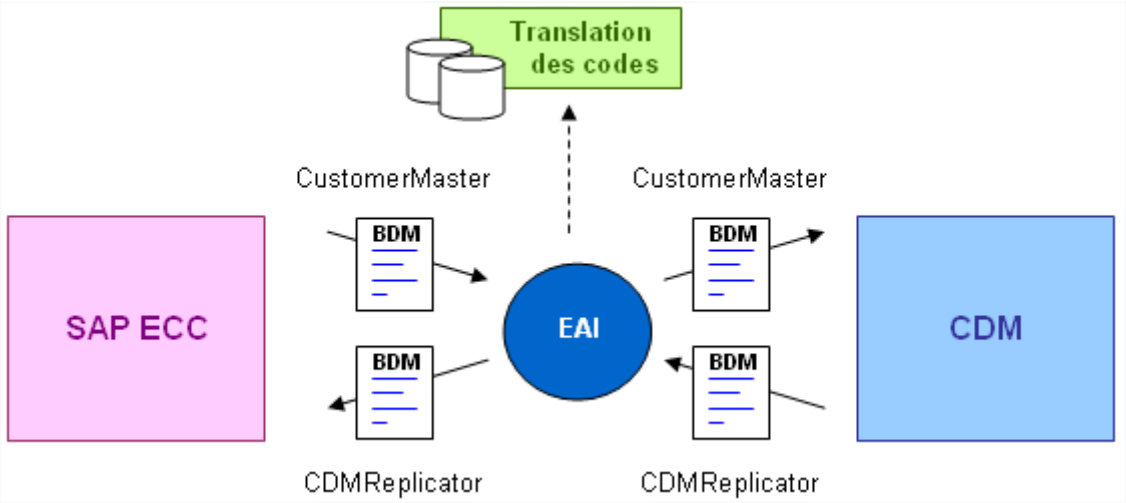


Illustration 29: Architecture de la solution, publication et souscription des BDM « CustomerMaster » et « CDMReplicator ».

5.3 L'interface de transfert des clients

Les clients sont créés dans SAP à partir de la transaction VD01. Les transactions VD02 et VD03 permettent quant à elles de respectivement modifier et afficher les données sur les acheteurs. La création d'un nouveau client requiert uniquement de compléter les informations d'entête parmi lesquelles le nom, l'adresse et le type du compte. Un compteur interne à SAP affectera automatiquement un numéro unique au client afin de pouvoir l'identifier.

A ce stade, le transfert des clients vers CDM est encore prématuré car ils ne disposent pas encore d'extension de vente. C'est en effet uniquement lorsque la première extension sera créée - et pour tous les futurs changements des informations - que l'interface devra se déclencher. Néanmoins, le

BDM d'extraction des clients de SAP ayant une importante probabilité d'être réutilisé pour les besoins d'autres applications, il est préférable d'implémenter les filtres au plus tard dans le processus technique afin de ne pas compromettre les futurs usages. Ainsi, peu importe si le client dispose d'extensions de vente ou pas, chaque création ou changement déclenchera l'interface et le connecteur de souscription de CDM se chargera lui même de filtrer les données.

SAP ne permet pas d'effacer des clients de sa base de données. Une case à cocher au niveau des données d'entête, ainsi qu'une autre sur chaque extension de vente permettent néanmoins de procéder à un effacement virtuel aussi appelé « effacement doux ». La validation de ces cases doit être répliquée sur CDM de telle manière que si la case située au niveau de l'entête est cochée, toutes les extensions de vente seront considérées comme effacées. En revanche, si seulement une case d'effacement doux disposée au niveau d'une extension de vente d'un client est cochée, seule cette extension sera considérée comme détruite.

5.3.1 Structure du BDM « CustomerMaster »

La structure du BDM « CustomerMaster » est basée sur l'organisation des données clients dans SAP. Ainsi le format comporte deux structures de base, une pour les données principales puis une seconde pour les informations spécifiques aux sites de vente. Cette dernière inclut de plus une sous structure itérative identifiant les partenaires (clients de livraisons, commerciaux...) assignés à la vue de vente du client.

- Informations générales sur le client (CustomerHeader), structure non itérative:
  Numéro de client, type de compte, nom du client, lignes d'adresse 1,2 et 3, code postal, ville, cote état, pays, numéro de téléphone, numéro de fax, clef de recherche, langage premier, indicateur d'effacement général.

- Informations spécifiques aux sites de vente (CustomerData), structure itérative:
  Code du site de vente, code de la chaîne de distribution, division, code société, date de création, moyen de transport, type, contact principal.

- Partenaires des sites de vente (Partners), structure itérative pour chaque site de vente:
  Code du type de partenaire, numéro du partenaire.

Par ailleurs afin de faciliter le suivi des messages entre SAP et WebMethods, le BDM inclut une structure technique appelée « EventIdentification » qui comporte un unique champ nommé « IDocNumber ».
A chaque changement des données sur un client de type tiers, l'ensemble des champs listés ci-dessus doivent être extraits puis publiés dans le BDM. A la réception, CDM se synchronisera en identifiant les changements au travers d'une comparaison des données de sa base avec les données reçues depuis l'interface. Le choix de publier des BDM complets sans prendre en compte uniquement les zones modifiées permet de ne pas contraindre la réutilisation de l'interface.

La spécification de mapping du BDM « CustomerMaster » comporte deux onglets, un premier pour définir les règles de publication depuis l'iDoc créé dans SAP et un second décrivant la logique d'intégration dans l'application CDM.

5.3.2 Conception SAP

La conception d'une interface exportant des données de SAP suit un processus organisé. Dans un premier temps il faut déterminer dans quel module les données sont enregistrées puis lister les principales tables sous jacentes. Il est ensuite nécessaire de déterminer le type d'iDoc, c'est à dire le format de l'iDoc qui sera utilisé pour le transport des données. Le format choisi peut être soit standard, étendu d'un iDoc standard ou entièrement personnalisé. Une fois ce choix entériné, le mécanisme de déclenchement doit être choisi, puis éventuellement développé avec les règles de remplissage de l'iDoc. Enfin, la dernière étape dans SAP avant de procéder aux tests est de réaliser la configuration ALE, puis la mise en place des traitements automatisés en arrière plan.

5.3.2.1 Données sur les clients

Les données sur les clients sont contenues dans le module Sales and Distribution (SD) de SAP ECC. L'illustration ci-dessous représente un sous ensemble des données générales d'un client SAP nommé « AAA International Company », ayant pour numéro 10010384.

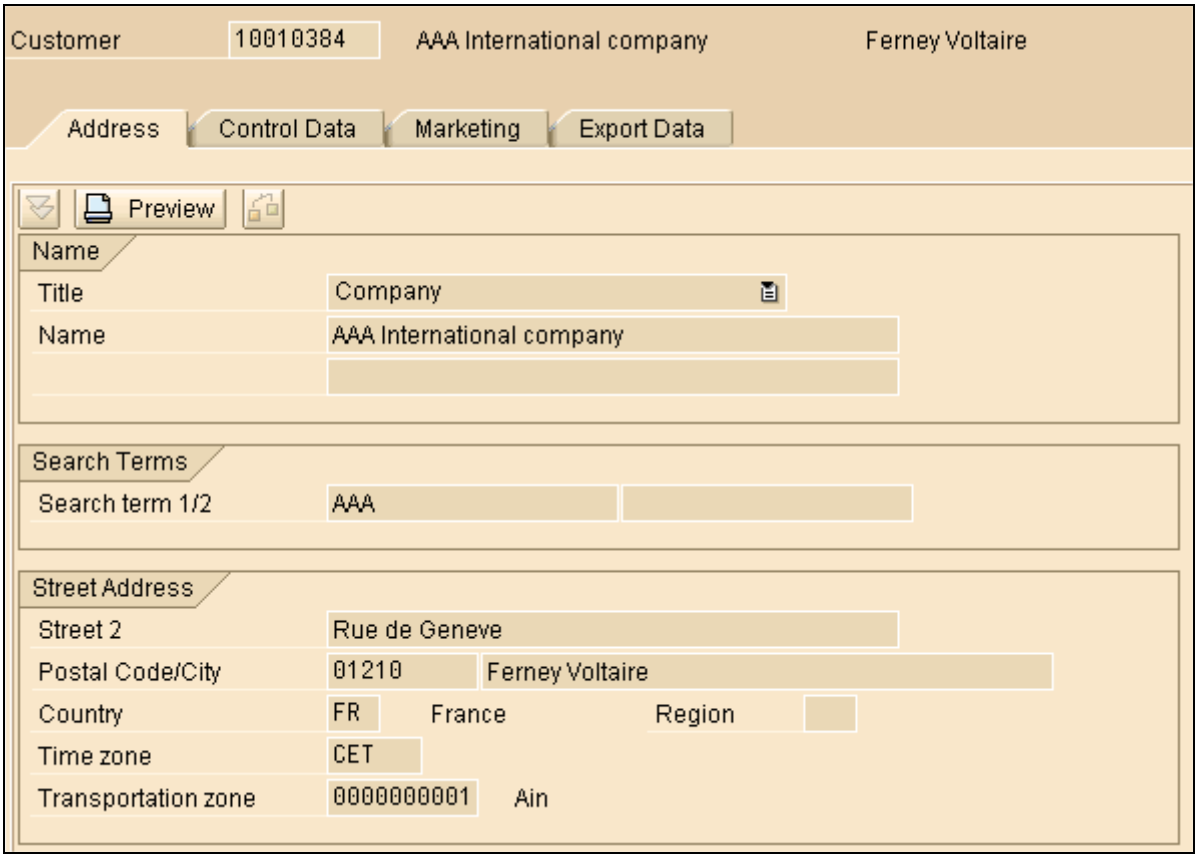


Illustration 30: Exemple de données générales d'un client SAP avec la transaction VD03.

Hormis leur adresse, les données générales sur les acheteurs sont enregistrées dans la table KNA1 dont la clef est simplement composée du numéro de client. Les adresses sont quant à elles enregistrées dans la table générale ADRC. Cette dernière, dont la clef est un numéro de séquence automatiquement généré comporte l'ensemble des adresses référencées dans les différents objets SAP tels que les clients, fournisseurs, partenaires, lieux de stockages et autres. Ainsi pour chaque client, un champ de la table KNA1 comporte la clef de son adresse stockée dans la table ADRC.

Les données spécifiques aux extensions de vente sont quant à elles stockées dans la table KNVV. Chaque site de vente est identifié par un code appelé « Sales Organization », un numéro de chaîne de distribution et un code de division pouvant être soit « FR » pour le segment des parfums ou « FL » pour le segment des arômes. En conséquence, la clef de la table KNVV est composée du numéro de client ainsi que de ces trois derniers champs. L'illustration ci-après représente un sous ensemble des données spécifiques du client 10010384 pour la chaîne de distribution 01 du site de vente « Givaudan Suisse Fragrances ».

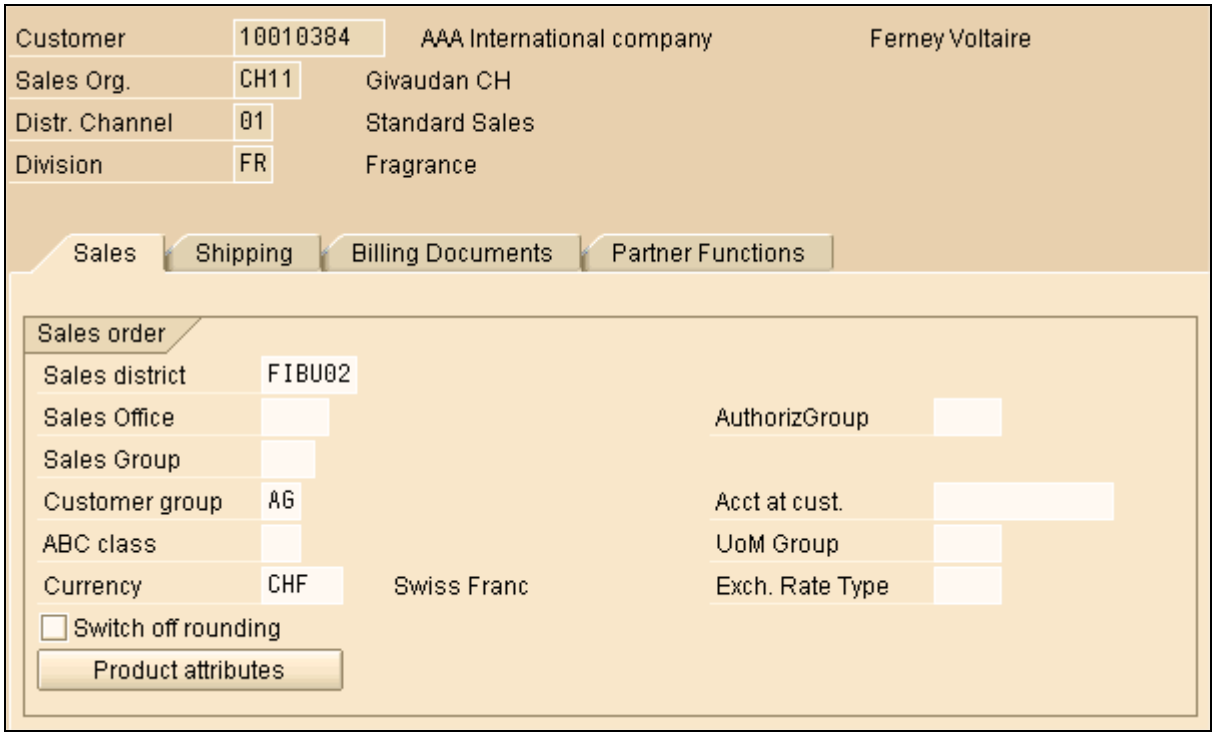


Illustration 31: Exemple de données de site de vente pour un client SAP avec la transaction VD03.

Les données sur les partenaires spécifiques à chaque extension de vente d'un client sont maintenues au travers de l'onglet « Partner Functions » de l'illustration ci-dessus. Chaque partenaire est enregistré dans la table KNVP dont la clef est composée des quatre zones identifiant le site de vente et le client, d'un code de type de partenaire ainsi que d'un compteur afin de permettre de disposer de plusieurs partenaires de même type.

5.3.2.2 Format de l'iDoc

SAP dispose d'un format d'iDoc standard pour l'extraction des données sur les clients. Bien que riche, ce format identifié par le type d'iDoc « DEBMAS06 » ne comporte pas l'ensemble des informations sur les adresses des clients qui sont requises pour publier le BDM « CustomerMaster ». Pour combler ce problème il est nécessaire de créer une extension du type d'iDoc « DEBMAS06 » dans laquelle un nouveau segment sera inclu pour contenir les données manquantes.

La première étape consiste à créer un nouveau type de segment au travers de la transaction WE31. Comme tout développement ou personnalisation non fournie par l'éditeur SAP AG, le nom du format du segment se doit de débuter par la lettre Z afin de le distinguer des composants standard. Intitulé « ZE1KNA1M », ce segment a été défini pour contenir les divers champs manquant de la table des adresses.

La seconde étape est de créer l'extension de l'iDoc standard puis d'y assigner le ou les nouveaux segments au niveau correspondant de la structure hiérarchique. Créé conformément à la convention de nommage des formats d'iDocs, l'extension « ZDEBMAS06A » comporte le nouveau segment « ZE1KNA1M » situé sous le segment « E1KNA1M » contenant les données générales du client issues de la table KNA1. L'illustration ci-après représente l'extension conçue au travers de l'outil de création des formats d'iDocs accessible par la transaction WE30.

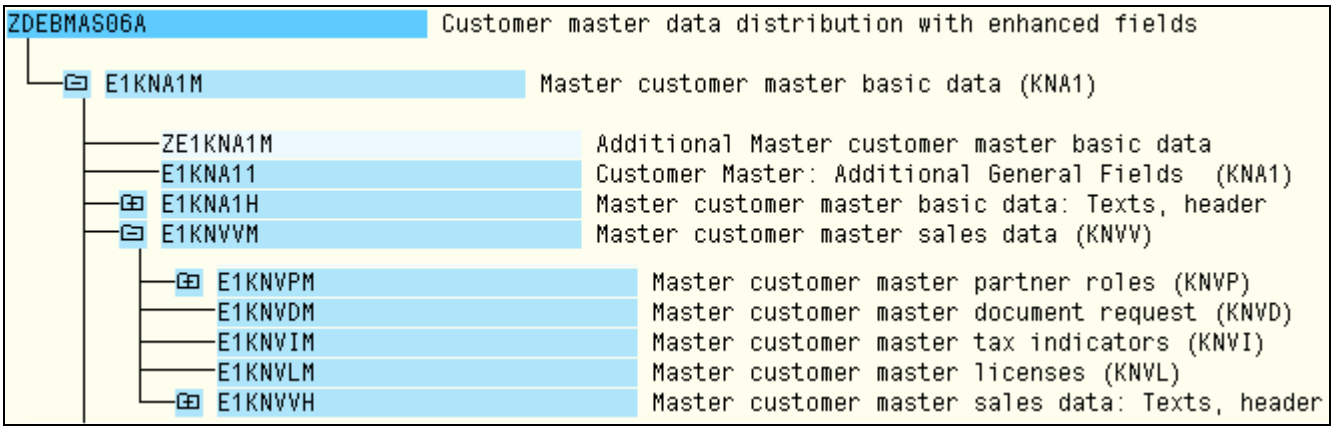
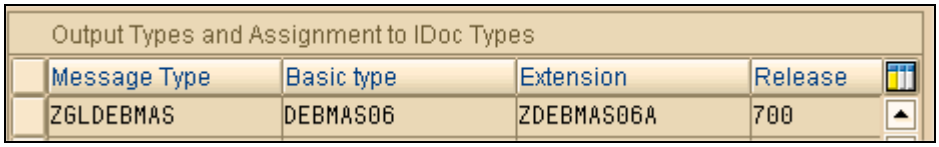


Illustration 32: Extension « ZDEBMAS06A » de l'iDoc standard « DEBMAS06 ».

L'extension comporte une multitude de segments transportant des données non utiles pour l'interface de synchronisation des clients avec CDM. Issus de l'iDoc standard « DEBMAS06 », ces segments peuvent être supprimés avec la transaction BD53. Néanmoins, le mécanisme de réduction des iDocs de SAP intervient uniquement en fin du processus lorsque les données sont déjà sélectionnées. Les bénéfices sur les performances sont donc nuls, mis à part pour le réseau dans le cas de transferts d'iDocs volumineux. Bien que n'entrant pas dans ce cadre, l'extension « ZDEBMAS06A » fut réduite afin que seuls les segments contenant les données générales (E1KNA1M, ZE1KNA1M), les extensions de vente (E1KNVVM) et les partenaires

(E1KNVPM) soient complétés lors de la création des iDocs. Les autres segments restent présents dans le format, mais ils seront toujours vides.

La dernière étape de préparation du format consiste à créer un type de message qui sera utilisé dans SAP pour la configuration ALE ainsi que pour la mise en place du mécanisme de déclenchement et d'affectation au BDM « CustomerMaster » dans l'EAI WebMethods. Le type de message « ZGLDEBMAS » doit être premièrement référencé avec un intitulé textuel dans la transaction WE81. La transaction WE82 sert ensuite à assigner le nouveau type de message au type d'iDoc et à son extension. L'illustration ci-dessous représente la configuration de cette affectation.


Output Types and Assignment to IDoc Types

Message Type	Basic type	Extension	Release
ZGLDEBMAS	DEBMAS06	ZDEBMAS06A	700

Illustration 33: Configuration des affectations du message type « ZGLDEBMAS » dans la transaction WE82.

La spécification d'une extension d'un iDoc standard, ou même la création d'un nouvel iDoc au travers des transactions WE30, WE31, BD53, WE81 et WE82 sont uniquement des étapes permettant de spécifier la structure des échanges. La logique de remplissage des segments non standards devra obligatoirement être développée en ABAP pour avoir des iDocs générés avec les informations complétées dans les nouvelles zones.

5.3.2.3 Mécanisme de déclenchement

Les changements des données client de SAP peuvent être détectés au moyen des pointeurs de modification. Bien que d'autres solutions soient possibles, l'usage des pointeurs fut sélectionné car il s'agit de la méthode préconisée sur le projet Outlook.

La configuration des pointeurs se fait au travers de la transaction BD52. Le mécanisme est simple: pour le type de message « ZGLDEBMAS » il suffit de lister l'ensemble des zones des tables qui une fois changées, effacées ou initialisées pour la première fois dans le cas d'un nouvel enregistrement, devront déclencher l'écriture d'un pointeur. C'est ainsi que pour cette interface, tous les champs des tables KNA1, KNVV, KNVP et ADRC nécessaires au remplissage du BDM « CustomerMaster » sont définis comme des déclencheurs dans la transaction BD52.

La seconde et dernière étape de configuration du mécanisme de déclenchement est de préciser dans la transaction BD60 le nom module de fonction ABAP qui devra être appelé pour lire les pointeurs et les marquer comme traités après avoir procédé à la création des iDocs correspondants. La fonction dans laquelle l'ensemble de ce code est développé pour l'interface d'extraction des clients est « Z_GL_OTC_IDOC_CREATEDEBMAS ». Sa conception est détaillée dans le chapitre suivant.

5.3.2.4 Développements

La première étape d'exécution de l'interface est de consulter la liste des pointeurs de modification ouverts pour le type de message « ZGLDEBMAS ». Si cette liste est vide, alors aucun traitement n'est à déclencher. En revanche s'il existe des pointeurs en attente de traitement, ils devront être utilisés pour déclencher l'interface.

Le module de fonction « Z_GL_OTC_IDOC_CREATEDEBMAS » précédemment configuré comme responsable du traitement des pointeurs pour le type de message « ZGLDEBMAS » sera appelé à chaque fois qu'une demande de traitement sera déclenchée par la transaction BD21 ou le programme RBDMIDOC. Le rôle de la fonction se limite à lire les pointeurs de modification, appeler le programme de création des iDocs pour chaque client concerné et marquer les pointeurs ouverts comme étant traités. Son algorithme est décrit ci-après.

```
Module de fonction Z_GL_OTC_IDOC_CREATEDEBMAS


// Lecture des pointeurs ouverts
Liste_pointeurs[] = Appel fonction 'CHANGE_POINTERS_READ' avec type de message
'ZGLDEBMAS'


Trier liste_pointeurs[] en fonction du nom de la table changée (KNA1, KNVV,
KNVP, ADRC) et de la clef des tables


Créer tableau liste_clients[], initialisé vide


// Déterminer les numéros des clients concernés
Boucle sur liste_pointeurs[] index i


      Si liste_pointeurs[i].table = 'KNA1', 'KNVV', ou 'KNVP'
            Vérifier si le numéro de client est dans le tableau
            liste_clients[]
            Si oui : ne rien faire
            Si non : ajouter numéro de client à liste_clients[]


      Sinon si liste_pointeurs[i].table = 'ADRC'
            Rechercher dans la table KNA1 quel est le numéro de client
            correspondant au numéro de l'adresse (clef de la table ADRC)
            Si aucun résultat : Ne rien faire. L'adresse ne correspond pas à
            un client mais à un autre type d'objet SAP (fournisseur, lieu de
            stockage...)
            Si résultat : Vérifier si le numéro de client est dans le tableau
            liste_clients[].
                  Si oui : ne rien faire
```

```
                Si non : ajouter numéro de client à liste_clients[]
            Fin si
        Fin si

Fin de la boucle

// Création des iDocs au travers de la fonction standard
Boucle sur liste_clients[] index i
      Appel fonction 'MASTERIDOC_CREATE_DEBMAS' avec numéro client
      liste_clients[i], extension 'ZDEBMAS06A' message type 'ZGLDEBMAS'
Fin boucle

// Fermeture des pointeurs
Appel fonction 'CHANGE_POINTERS_WRITE' avec tableau liste_pointeurs[] et type
de message 'ZGLDEBMAS'

Fin du module de fonction
```

La dernière étape de développement dans SAP est ensuite d'écrire le code ABAP nécessaire pour remplir le segment additionnel « ZE1KNA1M » de l'extension « ZDEBMAS06A ». Dans l'algorithme ci-dessus, la création des iDocs est prise en charge par le module de fonction standard « MASTERIDOC_CREATE_DEBMAS ». Bien qu'il soit impossible de modifier les développements réalisés par l'éditeur SAP AG, ces derniers incluent parfois des zones libres appelées « user exits » dans lesquelles il est possible d'ajouter du code personnalisé. Ainsi, les modules de fonctions SAP chargés de créer des iDocs comportent pour la plupart une zone de code libre située entre la fin de la préparation de l'iDoc standard et son enregistrement dans le système. C'est à cet endroit que le code de remplissage des segments additionnels doit être écrit. L'algorithme suivant décrit la portion de code ajoutée dans la zone libre du module de fonction « MASTERIDOC_CREATE_DEBMAS ».

```
Module de fonction MASTERIDOC_CREATE_DEBMAS

* Paramètres d'entrée: numéro_client, extension, type_message
*
* Programmation chargée de créer un iDoc au format standard DEBMAS06.
* Ce code n'est modifiable que par SAP AG.
*
* Contenu de l'iDoc standard dans tableau iDoc_contenu[] chaque ligne du
* tableau représente un segment de l'iDoc.
*
```

```
*** Début du user-exit DEBMAS: il est uniquement possible d'ajouter du code
personnalisé entre le début et la fin de ce bloc ***

Si type_message = 'ZGLDEBMAS' et extension = 'ZDEBMAS06A'
      Var i = 0
      Tant que iDoc_contenu[i].nom_segment != 'E1KNA1M'
            et i < taille du tableau iDoc_contenu[]

            adrc_clef = ABAP SQL: SELECT clef adresse FROM TABLE KNA1 WHERE
                        numéro_client = numéro_client (variable d'entrée de la
                        fonction)
            structure segment_ze1kna1m = ABAP SQL: SELECT champs adresse FROM
                        TABLE ARDC WHERE  clef_adresse = adrc_clef (numéro
                        d'adresse sélectionné dans la table KNA1)

            Ajouter structure segment_ze1kna1m au tableau iDoc_contenu[] juste
            après la ligne numéro i

            i = i + 1 // Incrémentation de l'index de la boucle
      Fin tant que
Fin Boucle

*** Fin du user-exit DEBMAS: il n'est plus possible d'ajouter du code
personnalisé au dessous de ce bloc ***

Vérification filtres auprès du modèle de distribution ALE
Appel fonction IDOC_CREATE avec tableau iDoc_contenu[]

Fin du module fonction
```

L'écriture des programmes et modules de fonctions s'effectue au travers de l'environnement de développement ABAP accessible par la transaction SE80.

5.3.2.5 Configuration ALE

La configuration ALE d'une interface sortante de SAP s'exécute en deux étapes. La première consiste à assigner dans la transaction WE20 le type de message et son extension à un profil de partenaire qui sera utilisé pour la communication avec l'intergiciel WebMethods. L'interface d'extraction des clients est configurée sur le profil de partenaire « K1_OTC », tel qu'illustré ci-après. Le port « WM_K1_OTC » identifie une connexion avec l'EAI tandis que la taille de paquet configurée à 25 signale que dans une même communication réseau, au maximum 25 iDocs de type « ZGLDEBMAS » seront envoyés. S'il y a plus d'iDocs à envoyer, alors le transfert sera réalisé par groupe de 25 iDocs. Enfin, non-cochée la case « Collect IDocs » signale que les iDocs ne seront pas envoyés automatiquement à la couche réseau.

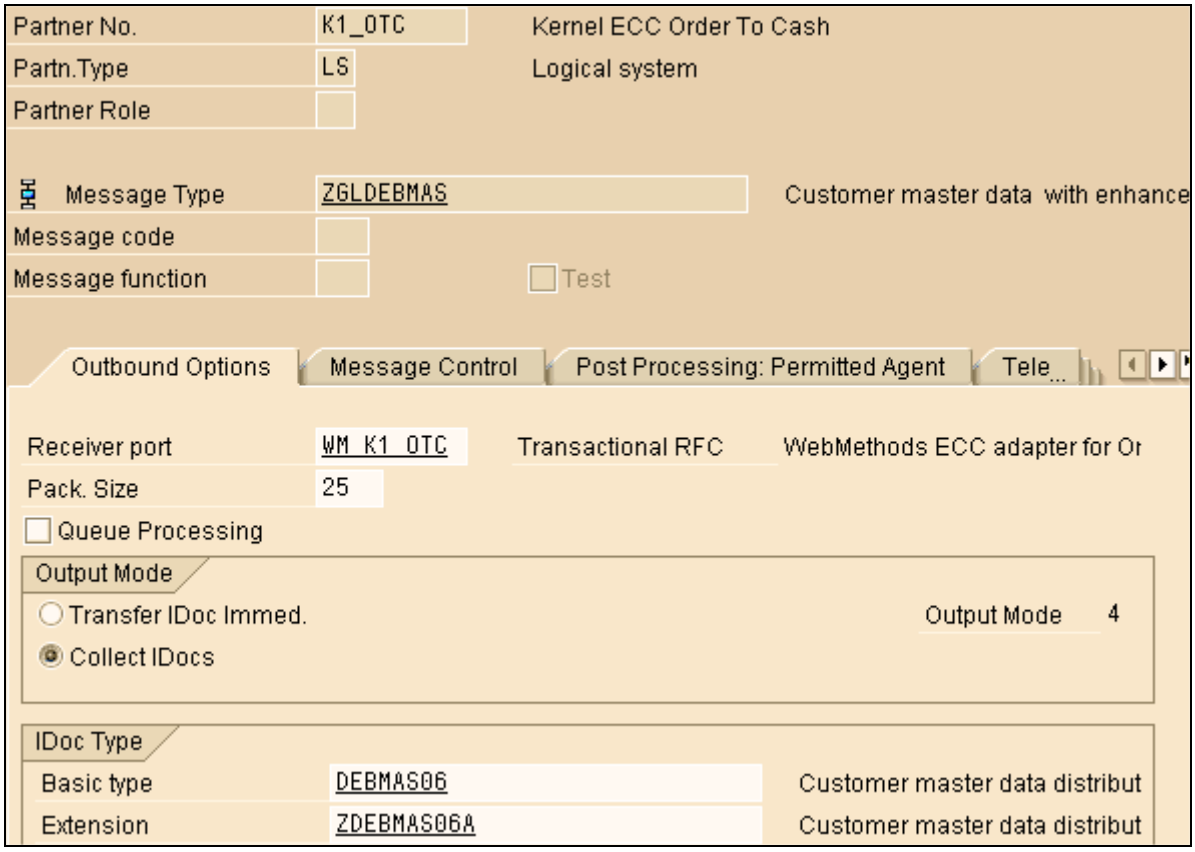


Illustration 34: Configuration du profil de partenaire pour le type de message « ZGLDEBMAS ».

La deuxième étape de la configuration ALE est de définir les filtres dans le modèle de distribution. Selon les mécanismes standard de SAP mais malgré qu'aucune règle générale n'existe, la définition du modèle de distribution n'est pas toujours obligatoire ni utile. Dans le cas de l'interface d'extraction des clients son usage est exigé par le contrôle du modèle imposé dans la partie de code standard de la fonction « MASTERIDOC_CREATE_DEBMAS », juste avant l'appel de la fonction « IDOC_CREATE ». Le modèle se configure au travers de la transaction BD64. L'illustration ci-dessous représente la configuration du modèle de distribution pour le type de message « ZGLDEBMAS ». Un filtre stipule que seuls les clients de type tiers (code Z001) devront être extraits au travers de cette interface.

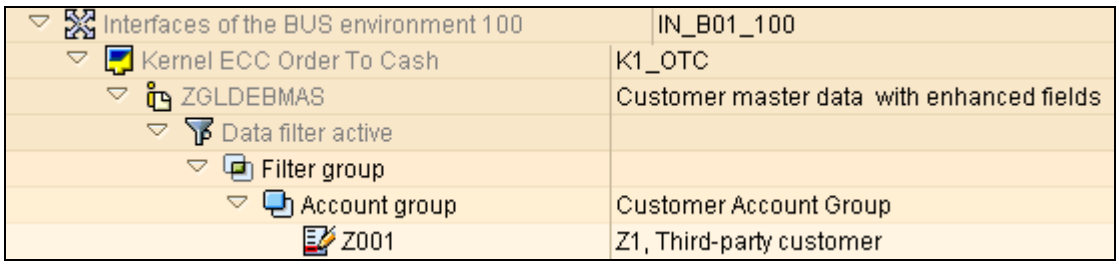


Illustration 35: Configuration du modèle de distribution et du filtre client-tier pour le type de message « ZGLDEBMAS ».

5.3.2.6 Traitements automatisés

Pour permettre à l'interface de fonctionner de manière automatisée, il est nécessaire de mettre en place deux traitements en arrière plan qui seront déclenchés à fréquence régulière.

Le premier traitement est celui qui déclenchera la lecture des pointeurs de modification et la création des iDocs. Il se déroule en exécutant le programme « RBDMIDOC » avec le nom du type de message « ZGLDEBMAS » dans l'unique zone de saisie de l'écran de sélection. Le second consiste quant à lui à exécuter le programme « RSEOUT00 » afin de transmettre les iDocs vers la couche réseau et ainsi les faire passer du statut 30 « iDoc en attente de transfert » à 03 « iDoc transféré à la couche communication ». Les déclenchements de ces programmes sont configurés via la transaction SM37. Ils disposent chacun d'une variante dédiée comportant leurs paramètres.

5.3.3 Conception WebMethods

Les développements WebMethods sont réalisés en trois parties:

- Développement de la structure du BDM: pré requis pour les deux autres étapes de développement WebMethods, la conception de la structure requiert que sa définition soit terminée dans le document de spécification au format Microsoft Excel.

- Développement de la logique de publication: la structure du BDM doit être préalablement conçue dans WebMethods ainsi que la structure de l'extension d'iDoc « ZDEBMAS06A » dans SAP. En effet, la conception des interfaces dans l'intergiciel WebMethods repose sur des mécanismes d'introspection dans les applications. Tant que les structures, iDocs ou tables utilisées n'existent pas, il n'est pas possible de concevoir des mappings.

- Développement de la logique de souscription: la conception de la structure du BDM et des tables de réception des données dans CDM doivent être préalablement terminées, selon les mêmes raisons que pour le développement de la logique de publication. Par ailleurs les règles de translation des codes SAP en codes BPCS s'appliquant au moment de la souscription, les tables de correspondance doivent elles aussi être disponibles.

Les mappings se développent rapidement grâce à un mécanisme de conception graphique. Les correspondances entre les champs se font en les sélectionnant un à un puis en les glissant jusqu'aux zones cibles, qui peuvent provenir soit de tables, d'iDocs, de BDM, de structures locales ou autres. Des opérateurs sont de plus disponibles afin de réaliser des opérations telles que des choix (bloc if) ou des boucles (bloc loop).
Dans la plupart des cas, le développement d'une nouvelle interface sur WebMethods se termine en moins de cinq jours de temps cumulé. Ainsi, pour limiter les problèmes de coordination au cours des tests il est préférable que pour une même interface les trois parties de la réalisation soient conçues par le même développeur.

5.3.3.1 Logique de publication SAP ECC

La spécification des règles de publication est réalisée au moment de la phase d'analyse de la structure de l'iDoc à concevoir. Ce travail se fait en collaboration entre les développeurs ABAP, WebMethods et l'architecte d'intégration spécialiste des interfaces. L'illustration ci-après représente les règles de publication du BDM « CustomerMaster » spécifiées dans l'onglet « SAP ECC Pub » du document de mapping. Le processus WebMethods est déclenché dès lors qu'un iDoc avec le type de message « ZGLDEBMAS » est reçu.

					MT ZGLDEBMAS, EXT ZDEBMAS06A	
Structure	Zone	Iteratif	Taille	Type	Segment	Zone / Logique
	IDocNumber				EDI_DC40	DOCNUM
CustomerHeader		N				
	Customer Code		10	String	E1KNA1M	KUNNR
	Customer Name		35	String	E1KNA1M	NAME1
	Address Line 1		40	String	ZE1KNA1M	STREET2
	Address Line 2		40	String	ZE1KNA1M	STREET3
	Address Line 3		40	String	ZE1KNA1M	STREET4
	Postal Code		10	String	E1KNA1M	PSTLZ
	City		30	String	E1KNA1M	ORT001
	State Code		3	String	ZE1KNA1M	REGIO
	Country Code		3	String	E1KNA1M	LAND1
	Alpha Search Key		40	String	ZE1KNA1M	SORT1
	Primary Language		1	String	E1KNA1M	SPRAS
	Fax Number		31	String	ZE1KNA1M	TELFX
	Telephone Number		16	String	ZE1KNA1M	TELF1
	Account Group			String	E1KNA1M	Z001
	Logical Delete			String	E1KNA1M	If E1KNA1M-NODEL = 'X', then value must be 'Y', else empty
CustomerData		Y				
	Sales organization		4	String	E1KNVVM	VKORG
	Division		2	String	E1KNVVM	SPART
	Distribution channel		2	String	E1KNVVM	VTWEG
	ABC Code		2	String	E1KNVVM	KLABC
	Date created		8	Date	E1KNVVM	ERDAT
	Logical Delete		1	String	E1KNVVM	If E1KNVVM-LOEVM = 'X', then Logical Delete must be 'Y'
	Means of Transportation			String	E1KNVVM	VSBED
	Partner Function	Y				
			2	String	E1KNVPM	PARVW
			10	String	E1KNVPM	KUNN2

Illustration 36: Règles de publication du BDM « CustomerMaster ».

5.3.3.2 Logique de souscription CDM

La logique de souscription par CDM au BDM « CustomerMaster » doit inclure toutes les règles permettant de s'assurer que les informations reçues sont bien celles attendues. Ainsi, le filtre de vérification du type de compte doit être répliqué dans WebMethods car il est probable que celui ci soit un jour supprimé du modèle de distribution de SAP afin de permettre la réception

de tous types de clients par d'autres applications. Si ce filtre n'est pas répliqué, alors le risque est d'un jour envoyer des données invalides à CDM.
Par ailleurs, conçu de manière ouverte le BDM d'extraction des clients est publié pour tous les changements de données sur les acheteurs tiers de SAP. Or, CDM ne doit recevoir les informations que si les clients disposent d'au moins une extension de vente.

La réception des données dans CDM se fait au travers de deux tables spécialement créées pour les besoins de cette nouvelle interface. La première table, CDMIBT doit contenir un nouvel enregistrement pour chaque extension de vente du client reçu. Les champs concernés couvrent la partie générale du BDM ainsi que la partie dédiée aux extensions de vente. La seconde table, CDMIPF, doit être alimentée d'un nouvel enregistrement pour chaque partenaire assigné au site de vente. Enfin à la fin des insertions dans ces deux tables, un programme doit être appelé sur le système AS400 qui héberge CDM afin de déclencher la phase finale du processus d'intégration dans l'application. La logique de translation des codes s'effectue en amont de toutes ces opérations. En cas d'erreur au cours de ce traitement, alors la transaction est rejetée. Elle pourra être ré-exécutée à partir de l'outil de supervision de WebMethods. L'illustration ci-dessous représente un extrait du travail de mapping réalisé dans l'EAI pour la souscription du BDM « CustomerMaster ».

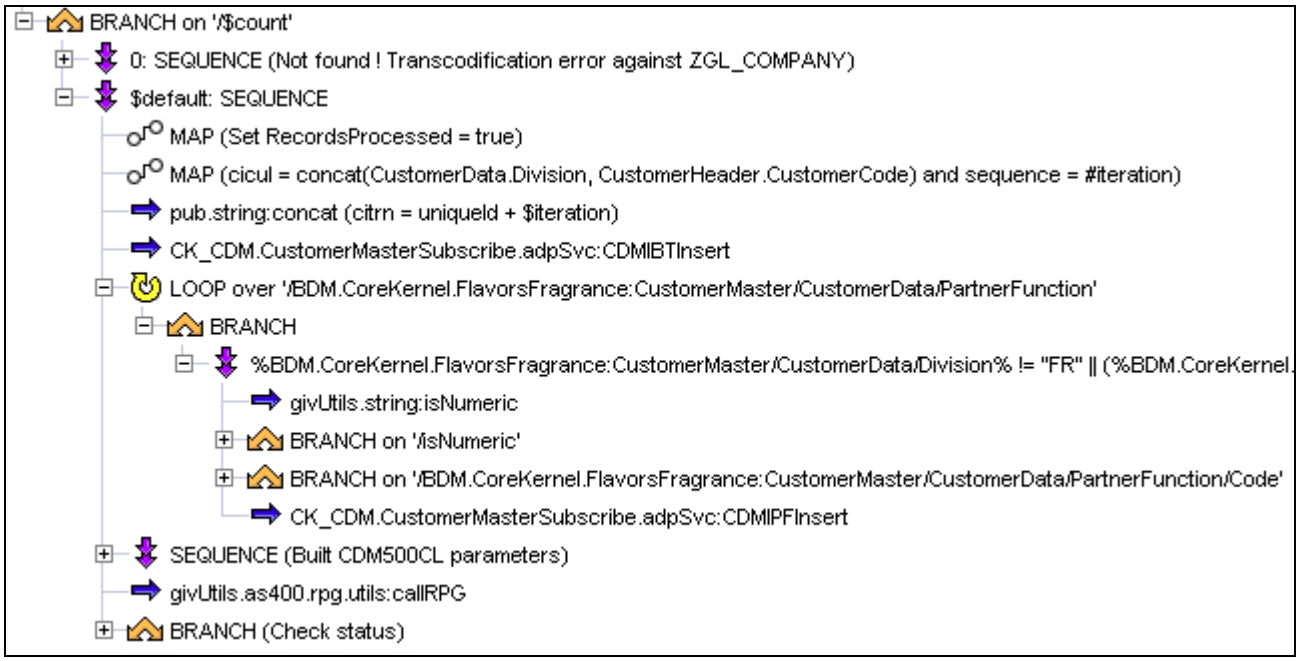


Illustration 37: Extrait du développement WebMethods pour la souscription au BDM « CustomerMaster »

5.3.4 Conception CDM

Une fois reçus dans CDM, les clients doivent être traités de manière similaire aux mécanismes d'intégration existants avec BPCS. Une adaptation fut réalisée à ce niveau afin de permettre une classification automatique dans la hiérarchie des clients reçus de SAP ECC pour lesquels d'autres extensions de vente ont déjà été classifiées. La mise en place d'un tel mécanisme avec l'ancien progiciel de gestion intégré n'était pas possible du fait que chaque client disposait

d'un numéro différent dans chaque instance de BPCS. Commandités par le projet Outlook, les développements des adaptations sur CDM ont été pris en charge par une organisation séparée exerçant un support sur l'application depuis de nombreuses années.

5.4 L'interface de transfert des hiérarchies

L'interface de transfert des hiérarchies dispose de nombreuses contraintes liées à la conception de CDM. Le projet Outlook doit par ailleurs recevoir les données au travers du BDM existant « CDMReplicator » en évitant toute modification de ses règles de publication afin de ne pas briser l'écosystème de plus de dix applications recevant la hiérarchie au travers de ce canal.

Développé dans les années 1990, CDM a été conçu pour répondre aux contraintes des environnements BPCS multi-instances. Dans ce contexte, les clients créés dans chacun des sites disposent de numéros d'identifiants générés par des compteurs différents. Ainsi, un même client utilisé sur deux sites Givaudan dispose pour chacun d'un numéro d'identifiant dédié. Pour combler ce problème, un niveau hiérarchique appelé « Unique » a été créé dans CDM afin de regrouper sous un même code toutes les instances d'un même client. Ainsi, la hiérarchie des clients sur CDM est composée de quatre niveaux tel qu'illustré ci-dessous.

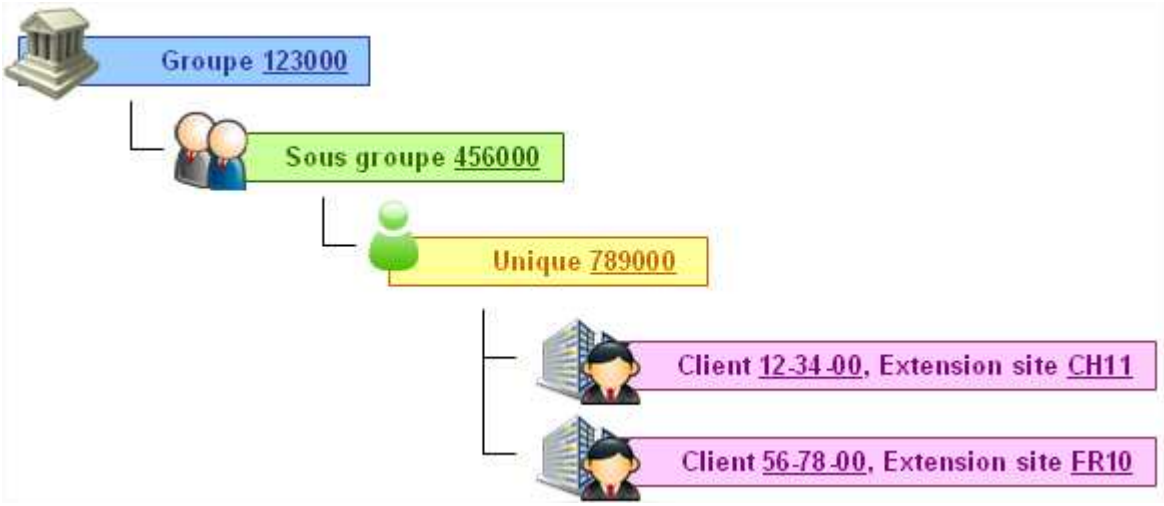


Illustration 38: Structure hiérarchique des clients dans CDM.

En implémentant SAP tel un système globalisé, un même client disposera toujours du même numéro quel que soit le site de vente où il est utilisé. En conséquence, le niveau hiérarchique « Unique » n'aura plus de sens une fois que le nouveau système sera productif sur tous les sites Givaudan. Ainsi, il fut choisi de ne pas répliquer dans la hiérarchie des clients de SAP ce niveau désormais inutile pour l'organisation métier. Néanmoins, le fonctionnement de l'interface « CDMReplicator » ne permet pas à SAP de faire entièrement l'impasse sur le niveau « Unique ». Il devra donc obligatoirement être modélisé pour des besoins techniques mais il sera entièrement caché aux utilisateurs.

5.4.1 Structure du BDM « CDMReplicator »

Conçu au cours de l'année 2005, le BDM « CDMReplicator » fonctionne de manière particulière. Chaque message publié dispose d'une entité signalant le type de son contenu. En fonction de l'entité, l'application réceptrice devra exécuter un traitement de synchronisation adapté qui sera aussi dépendant des messages précédemment reçus. Ainsi, la stabilité de l'interface repose sur l'obligation stricte à l'application réceptrice de traiter les messages dans l'ordre exact où ils ont été envoyés par CDM. Dans le cas où une erreur ne permet pas de traiter un message, alors l'interface doit être entièrement stoppée pour éviter le risque qu'une succession importante de transactions rejetées engendre une corruption de la hiérarchie dans le système récepteur. Les principales entités du BDM « CDMReplicator » et les actions que leur réception doit déclencher sont listées ci-après.

Entité groupe : CDMGRP

Synchronise un client de type groupe.
Le code action indique si le groupe doit être créé, mis à jour ou effacé.

Contenu:

- Type d'action (I: création, U: mise à jour, D: effacement)
- Numéro du client groupe
- Nom du client groupe

Entité sous groupe : CDMSUB

Synchronise un client de type sous-groupe en tant que fils d'un client groupe indiqué.
Le code action indique si le sous-groupe doit être créé, mis à jour ou effacé.

Contenu:

- Type d'action (I: création, U: mise à jour, D: effacement)
- Numéro du client parent (type groupe)
- Numéro du client sous groupe
- Nom du client sous groupe

Entité unique : CDMCUS

Pareil que l'entité CDMCUS mais pour gérer la relation d'un client de type unique avec un client de type sous-groupe.

Entité locale : CDMRCM

Synchronise le dernier niveau de la hiérarchie. L'affection de chaque extension de vente des clients tiers à un client de type unique est gérée au travers de cette entité.

Contenu:

- Type d'action (I: création, U: mise à jour, D: effacement)
- Numéro du client parent de type « unique »
- Numéro du client local
- Code de la division et de l'extension de vente du client local

Il existe par ailleurs d'autres entités utilisées par l'interface afin de transférer des attributs sur les affectations. Listées ci-dessous, ces dernières sont négligeables pour expliquer le cœur du processus de réplication, elles seront donc que partiellement évoquées.

- Entité d'attributs de groupe CDMDGC
- Entité d'attributs de sous groupe CDMDSC
- Entité d'attributs de client unique CDMDUC

Lors de la création d'un nouveau client de type groupe, sous groupe ou unique, l'entité d'attribut correspondante est publiée deux fois: une première pour les attributs de la division des arômes, et une seconde pour les attributs de la division des parfums.

5.4.2 Conception SAP

Tel que pour une interface d'extraction, la première étape de conception d'une interface d'importation de données dans SAP est d'analyser les transactions et structures des tables dans lesquelles les informations seront enregistrées. Viennent ensuite la conception du format de l'iDoc et le développement de la logique de traitement. Enfin, la mise en place de la configuration ALE et des traitements automatisés termine la préparation de l'interface.

5.4.2.1 Données sur les hiérarchies

Dans le progiciel SAP ECC, la hiérarchie des clients peut être visualisée et maintenue via la transaction VDH2N. Toutes les données sont enregistrées dans la table KNVH où la validité des affectations est définie sur un intervalle de dates. L'illustration ci-après représente une hiérarchie de clients maintenue dans SAP. Pour faciliter leur distinction, les clients de type groupe et sous groupe sont préfixés par les lettres « G » et « S », puis doublé par division afin de satisfaire des contraintes spécifiques de gestion des données dans SAP. Seule la hiérarchie de la division des parfums (FR) dispose de clients au niveau le plus bas car les deux extensions de vente créées sur le client tiers 10010384 sont toutes deux assignées à des sites de vente de ce segment de marché.

Cust. hierarchy	Customer no.	Loc	Sales area
GRP-EXAMPLE	G017370	99-	XX10/01/FL
SGRP-EXAMPLE	S017066	99-	XX10/01/FL
GRP-EXAMPLE	G017370	99-	XX10/01/FR
SGRP-EXAMPLE	S017066	99-	XX10/01/FR
AAA International company	10010384	FR-01210...	FR10/01/FR
AAA International company	10010384	FR-01210...	CH11/01/FR

Illustration 39: Hiérarchie des clients SAP visualisée dans la transaction VDH2N.

Pour maintenir dans SAP une telle hiérarchie alimentée par le BDM « CDMReplicator », il est nécessaire de concevoir une solution permettant de maintenir une relation entre les clients de type tiers, unique et sous groupe. La solution adoptée consiste à créer un nouveau champ dans la table KNA1 qui enregistre les données générales des clients. Pour chaque client-tiers, cette nouvelle

zone contiendra le numéro de client unique auquel il est associé dans CDM. Enfin, l'affection des clients uniques aux sous groupes devra être enregistrée dans une nouvelle table dans laquelle le code numéro du sous groupe formera la clef.

5.4.2.2 Format de l'iDoc

La spécificité de la structure du BDM « CDMReplicator » ne permet pas l'usage d'un iDoc standard. Un iDoc entièrement dédié doit donc être conçu avec un format similaire à celui du BDM. La structure de ce nouveau type d'iDoc nommé « ZGLCUSHIER01 » est illustrée ci-dessous.

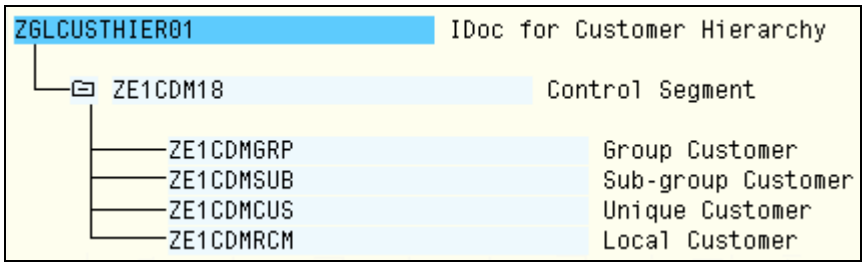


Illustration 40: Structure du type d'iDoc ZGLCUSTHIER01.

Le segment d'entête « ZE1CDM18 » comporte une zone qui devra être renseignée au moment du mapping dans WebMethods avec le nom de l'entité, puis une seconde indiquant le code du type d'action « I », « U » ou « D » selon qu'il s'agisse d'un nouveau client, d'une mise à jour ou d'un effacement. Enfin, chacun des sous segments de « ZE1CDM18 » est spécifique à un type d'entité. Uniquement l'un d'eux sera complété à la fois.

Les segments et le type d'iDoc sont créés avec les transactions WE30 et WE31 tel que vu précédemment pour l'interface d'extraction des clients. Le type de message « ZGLCUSTHIER » doit ensuite être défini dans la transaction WE81, puis assigné au type d'iDoc « ZGLCUSTHIER01 » avec la transaction WE82.

La dernière étape de préparation consiste à créer un code de traitement dans la transaction WE42. Renseigné au moment de la configuration ALE, ce code permet de déterminer quel est le module de fonction qui sera chargé d'exécuter le traitement de l'iDoc. L'illustration ci-dessous représente la configuration du code de traitement « ZLOCUSTHIER » avec le module de fonction « Z_GL_LO_F_IDOC_INPUT_CUSTHIER ».

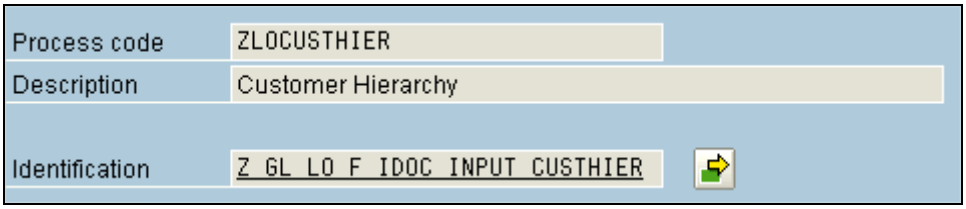


Illustration 41: Configuration du code de traitement à travers la transaction WE42.

5.4.2.3 Développement

La première étape du développement est de concevoir les nouvelles structures de données. Il s'agit donc ici de créer une zone dans la table KNA1 pour contenir le code du client de type unique, puis une nouvelle table ZGL_OTC_CDMUNSUB pour enregistrer les affectations entre les clients de type sous-groupe et unique. Ces deux tâches de développement se réalisent au travers de la transaction de gestion des structures de données SE11.

- Nouveau champ ZUNIQUE format numéro de 6 chiffres dans la table KNA1
- Nouvelle table ZGL_OTS_CDMUNSUB,
  - Champ ZUNIQUE format numéro de 6 chiffres. Clef de la table.
  - Champ ZSUBCUS format chaîne de 7 lettres. Ne peut pas être vide.

Une fois les structures de données prêtes, le développement en langage ABAP du module de fonction « Z_GL_LO_F_IDOC_INPUT_CUSTHIER » peut commencer. Ce travail se décompose en deux parties. Premièrement l'écriture du code de l'algorithme principal responsable de traiter les différentes entités en fonction des types d'action, puis le développement des sous fonctions de service chargées de mettre à jour la hiérarchie dans le système.

L'algorithme de traitement des entités est détaillé ci-dessous. Par mesure de bonne pratique, les types d'actions « I », « U » et « D » sont toujours vérifiés. Ainsi la réception d'une demande de création pour un client de type groupe, sous groupe ou unique existant déjà dans le système déclenchera une modification de ce dernier. A l'inverse, si une demande de mise à jour est reçue pour un client n'existant pas, alors il sera créé. Un tel procédé permet fiabiliser l'interface en évitant des erreurs inutiles.

```
Module de fonction  Z_GL_LO_F_IDOC_INPUT_CUSTHIER

* Paramètre d'entrée: tableau iDoc_contenu[] contenant une ligne pour chaque
* segment de l'iDoc ZGLCUSTHIER01 reçu

// Le premier segment de l'iDoc est obligatoirement « ZE1CDM18 »
code_action = Segment_entete[0].action
nom_entité = Segment_entete[0].entité
// Le second segment de l'iDoc est obligatoirement le segment d'entité à
traiter
Segment_entité = iDoc_contenu[1]

Si nom_entité = 'CDMGRP' // Synchronisation groupe

      code_groupe = 'G' + Segment_entité.numéro_client_groupe
      nom_groupe = Segment_entité.nom_client_groupe

      Si code_action = 'I' ou 'U' // création ou mise à jour
```

```
            Vérifier existence client groupe code_groupe
                  Si oui: modifier nom client avec nom_groupe
                  Sinon: créer client code_groupe avec nom nom_groupe
            // Création du client groupe dans la hiérarchie
            Appel fonction 'ZHIER_CREATE_GROUP' avec paramètre code_groupe
      Sinon si code_action= 'D' // effacement
            Vérifier existence client groupe code_groupe
                  Si oui: marquer le client groupe comme effacé
                  Sinon: ne rien faire
            // Effacement du client groupe dans la hiérarchie
            Appel fonction 'ZHIER_DEL_GROUP' avec paramètre code_groupe
      Fin si

Sinon si nom_entité = 'CDMSUB' // Synchronisation sous groupe

      code_sous_groupe = 'S' + Segment_entité.numéro_client_sous_groupe
      nom_sous_groupe = Segment_entité.nom_client_sous_groupe
      code_parent = Segment_entité.nom_client_groupe

      Si code_action = 'I' ou 'U' // création ou mise à jour
            Vérifier existence client groupe code_sous_groupe
                 Si oui: modifier nom client avec nom_sous_groupe
                 Sinon: créer client code_sous_groupe avec nom nom_sous_groupe
            // Création du client sous groupe dans la hiérarchie
            Appel fonction 'ZHIER_CREATE_SUBGROUP' avec paramètre
            code_sous_groupe et code_parent
      Sinon si code_action= 'D' // effacement
            Vérifier existence client groupe code_sous_groupe
                  Si oui: marquer le client groupe comme effacé
                  Sinon: ne rien faire
            // Effacement du client sous groupe dans la hiérarchie
            Appel fonction 'ZHIER_DEL_SUBGROUP' avec paramètre
            code_sous_groupe et code_parent
      Fin si

Sinon si nom_entité = 'CDMCUS' // Synchronisation client unique

      code_unique = Segment_entité.numéro_client_unique
      code_parent = Segment_entité.numéro_client_sous_groupe

      Si code_action = 'I' ou 'U' // création ou mise à jour
            Vérifier l'existence d'un enregistrement dans la table
```

```
            ZGL_OTC_CDMUNSUB pour client unique code_unique
                  Si oui: mettre à jour le record avec zsubcus = code_parent
                  Si non: insérer le record
            // Pas de mise à jour de la hiérarchie nécessaire à ce stade
      Sinon si code_action = 'D'
            Vérifier l'existence d'un enregistrement dans la table
            ZGL_OTC_CDMUNSUB pour client unique code_unique
                  Si oui: effacer le record
                  Si non: ne rien faire
            // Pas de mise à jour de la hiérarchie nécessaire à ce stade
      Fin si

Sinon si nom_entité = 'CDMRCM' // Synchronisation client local (client tiers
étendu à un site de vente)

      code_local = Segment_entité.numéro_client_local
      code_unique = Segment_entité.numéro_client_unique
      code_site_vente = Segment_entité.site_vente
      code_division = Segment_entité.division

      Enregistrer code_unique dans champ ZUNIQUE de la table KNA1 pour le
      client code_local
      Si code_action = 'I' ou 'U' // création ou mise à jour
            Vérifier l'existence du client local code_local  et de son
            extension de vente code_site_vente pour la division code_division
                  Si non: situation incohérente, générer une erreur et
                        terminer.
                  Si oui: lire code_sous_groupe dans ZGL_OTC_CDMUNSUB pour
                        clef code_unique
                  // Création/Mise à jour de l'affectation dans la hiérarchie
                  Appel fonction 'ZHIER_CREATE_LOCAL' avec paramètres
                  code_local, code_site_vente, code_division, code_sous_groupe
      Sinon si code_action = 'D'
            Vérifier l'existence du client local code_local  et de son
            extension de vente code_site_vente pour la division code_division
                  Si non: ne rien faire
                  Si oui: lire code_sous_groupe dans ZGL_OTC_CDMUNSUB pour
                        clef code_unique
                  // Effacement de l'affectation dans la hiérarchie
                  Appel fonction 'ZHIER_DEL_LOCAL' avec paramètres
                  code_local, code_site_vente, code_division, code_sous_groupe

Fin si
```

Le développement des sous fonctions de service consiste uniquement à formater des données pour correspondre aux besoins d'entrée des fonctions standard de mise à jour des hiérarchies dans SAP. D'une logique très simple à développer, ces dernières ne sont pas détaillées ici.

5.4.2.4 Configuration ALE

La configuration ALE d'une interface entrante consiste uniquement à compléter les profils de partenaires. Le modèle de distribution n'est en effet jamais utilisé pour les interfaces d'importation. L'illustration ci-dessous représente la configuration ALE pour le type de message « ZGLCUSTHIER ». Le code de traitement « ZLOCUSTHIER » sera utilisé par SAP pour identifier la fonction qui sera chargée de traiter les iDocs. L'option de déclenchement en arrière plan est par ailleurs cochée, ce qui est obligatoire pour la configuration du mécanisme de traitement en séquence stricte qui sera décrit dans le paragraphe suivant.

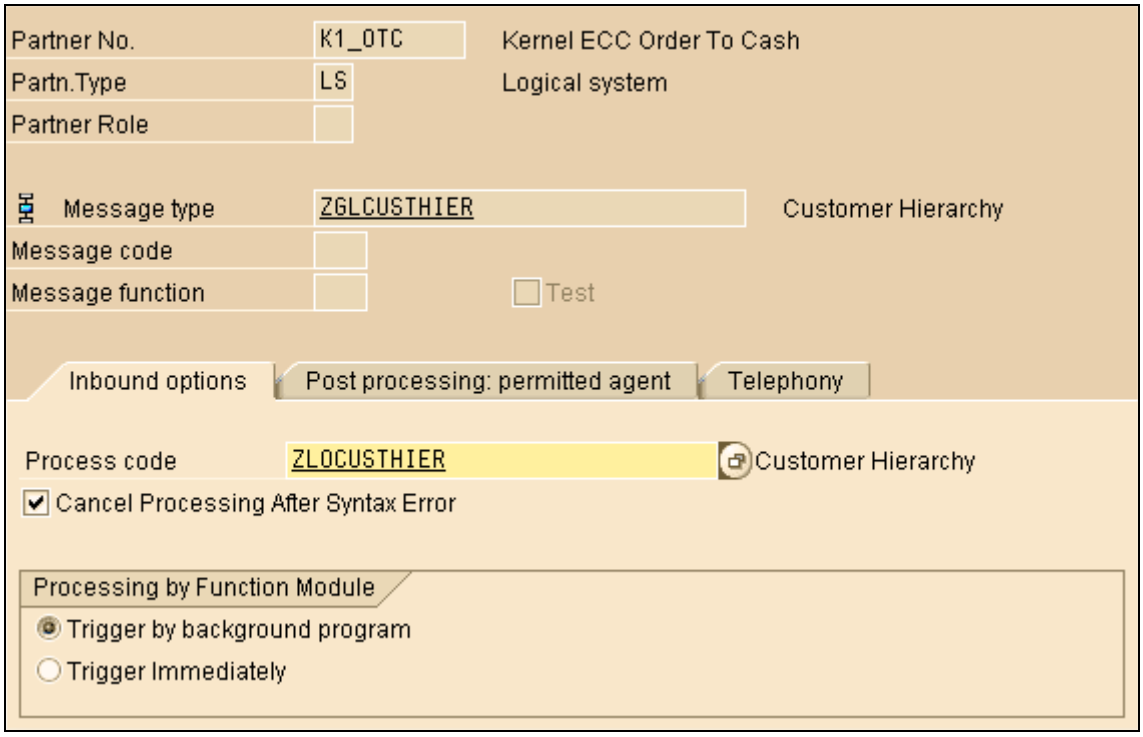


Illustration 42: Configuration du profil de partenaire pour le type de message « ZGLCUSTHIER ».

5.4.2.5 Traitements automatisés

Pour garantir un traitement des iDocs en séquence stricte, c'est à dire un arrêt complet de l'interface en cas de problème sur un message, il est nécessaire d'utiliser une file d'attente. La création des files d'attente est automatique, elle requiert uniquement à l'EAI qui transmet l'iDoc vers SAP de spécifier le nom de la file au moment du transfert. Si cette file n'existe pas, alors elle sera créée par le système. Une fois reçus, les iDocs ne seront pas disposés dans le l'état 64 « iDoc en attente de traitement » mais dans le statut 75 « iDoc en attente dans la file». A la fin de leur traitement, ils seront comme tous les autres iDocs en entrée de SAP, disposés dans le statut 53 « iDoc traité avec succès » ou 51 « iDoc exécuté avec erreur ».

Le traitement des iDocs disposés dans des files d'attente est déclenché par l'exécution du programme « RSEINBQUEUE ». Le seul paramètre requis pour son exécution est le nom de la file. Cette dernière peut aussi être administrée au travers de la transaction WEINBQUEUE, utilisée principalement dans les rares cas où il est nécessaire d'exclure un iDoc de la file.

5.4.3 Conception WebMethods

Du fait de la réutilisation de la partie publication de l'interface « CDMReplicator » et de la création d'un iDoc dédié au format semblable à la structure du BDM, les règles de mapping sont très faciles à développer dans l'EAI. Ce travail est illustré par la copie d'écran ci-dessous faite à partir de l'outil de développement WebMethods.

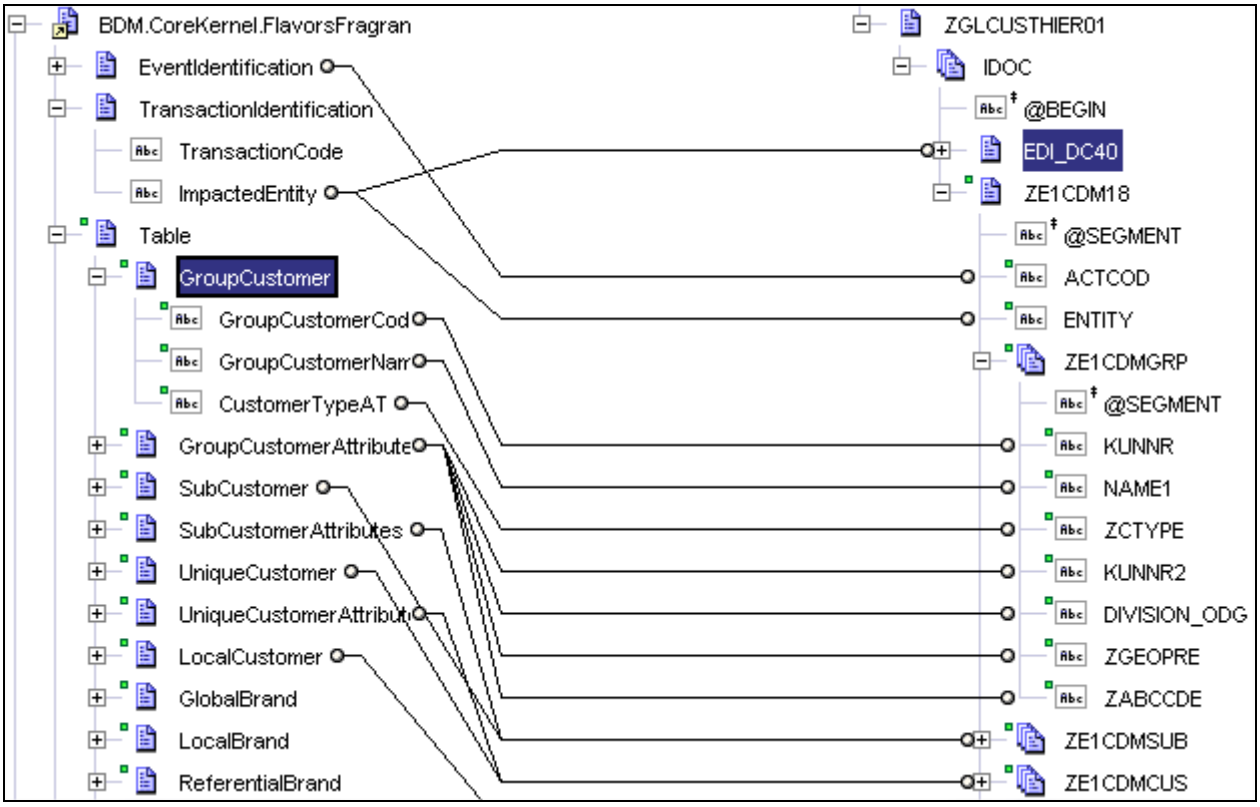


Illustration 43: Extrait du mapping de l'iDoc « ZGLCUSTHIER01 » à partir du BDM « CDMReplicator ».

Le service WebMethods chargé du transfert de l'iDoc vers SAP doit être configuré pour envoyer les iDocs dans une file d'attente nommée « IDOC_CUSTHIER ». Par ailleurs, CDM publiant une multitude d'entités non utiles pour la réplication des hiérarchies de clients dans SAP, il est nécessaire de disposer dans le connecteur de réception un filtre acceptant uniquement les entités suivantes:

- Entité groupe CDMGRP et attributs de groupe CDMDGC
- Entité sous groupe CDMSUB et attributs de sous groupe CDMDSC
- Entité client unique CDMCUS et attributs de client unique CDMDUC
- Entité client local (client de type tier) CDMRCM

5.5 Migration des données

Le système SAP étant initialement vide de données, il est nécessaire d'effectuer un chargement initial de la hiérarchie avant de pouvoir connecter le PGI à CDM. Cette migration peut se faire au travers de programmes qu'il faudra concevoir spécialement pour l'exécution de cette tâche, ou au travers de l'interface « CDMReplicator». Cette seconde option fut sélectionnée car elle permettait de fiabiliser l'interface par des tests en condition de charge massive et que la durée estimée, plus longue que lorsqu'un programme dédié est utilisé, n'engendrait pas de risque important pour le chemin critique du planning de démarrage.

La migration des données des clients et de leur hiérarchie se fait en deux phases. La première, appelée migration initiale doit se faire une seule fois dans le système afin d'initialiser les clients de type groupe, sous groupe et unique. Plus complexe, la seconde devra être répétée à chaque fois qu'un site sera migré de l'ancien progiciel de gestion intégré BPCS à SAP.

5.5.1 Migration initiale

En condition normale, l'interface « CDMReplicator» se déclenche uniquement lorsque des données sont créées, mises à jour ou effacées dans CDM. Pour l'exécution d'une telle migration, il va être nécessaire de forcer l'application à envoyer tous les éléments d'une série d'entités sans qu'ils aient subi de modification. Une simple requête SQL permet aux administrateurs de CDM d'exécuter cette opération. Néanmoins pour ne pas impacter les autres applications recevant déjà la hiérarchie des clients avec l'interface « CDMReplicator », ces dernières doivent être déconnectées à partir de l'EAI WebMethods le temps de l'exécution de la migration. Pour cette raison, l'accès aux utilisateurs doit aussi être bloqué afin d'éviter tout changement des données durant cette période.

Par ailleurs, le traitement dans SAP en séquence stricte des iDocs pose un problème double. Premièrement, s'arrêter à chaque erreur est irréaliste au vu du nombre conséquent d'iDocs à traiter. En second lieu, ce type de traitement est extrêmement long même si tous les iDocs peuvent être exécutés sans erreur à la première tentative. Ainsi, l'idée est de procéder aux traitements des messages de manière parallèle. Néanmoins étant donné la dépendance entre les messages, cela ne peut fonctionner que si les iDocs sont traités par groupe d'entités. Le tableau ci-dessous indique le volume de messages pour les entités qui devront être transférés à SAP lors de la réalisation de la migration initiale.

Entité principale	Type de clients	Nombre d'éléments	Entité d'attributs	Nombre d'éléments
CDMGRP	Groupes	24876	CDMDGC	49752
CDMSUB	Sous-groupes	25605	CDMDSC	51210
CDMCUS	Uniques	54399	CDMDUC	108996

Tableau II: Nombre de messages à transférer pour chaque entité.

Six étapes vont donc être nécessaires: tour à tour les iDocs de chaque entité vont être transférés à SAP, puis entièrement traités avant de procéder au transfert des messages de l'entité suivante. La configuration de l'envoi des iDocs dans la file d'attente « IDOC_CUSTHIER » doit être désactivée dans WebMethods. Le programme SAP « RSEINBQUEUE » chargé de traiter les files doit lui aussi être stoppé et temporairement remplacé par le programme « RBDAPP01 » dont la variante sera paramétrée de telle manière à activer les traitements en parallèle. Pour chaque entité, la durée du transfert des messages par l'EAI et des traitements des iDocs dans SAP est renseignée dans le tableau ci-dessous.

Ordre	Entité principale	Type	Nombre d'éléments	Durée du transfert des iDocs (EAI)	Durée du traitement des iDocs (SAP)
1	CDMGRP	Client groupes	24876	35 minutes	175 minutes
2	CDMDGC	Attributs clients groupes	49752	70 minutes	115 minutes
3	CDMSUB	Clients sous-groupes	25605	36 minutes	180 minutes
4	CDMDSC	Attributs clients sous groupes	51210	72 minutes	118 minutes
5	CDMCUS	Clients uniques	54399	76 minutes	32 minutes
6	CDMDUC	Attributs clients uniques	108996	152 minutes	40 minutes

Tableau III: Durée des transferts et traitements par entité.

La durée des transferts dans l'EAI WebMethods n'est pas dépendante de l'entité mais uniquement du nombre de messages. En effet pour cette interface, la complexité du traitement de mapping pour chaque type de transaction ne varie pas. En revanche, le traitement dans SAP diffère selon les entités. Ainsi « CDMGRP » et « CDMSUB » dont l'exécution requiert de créer des nouveaux clients dans le système, ont une durée de traitement plus longue que « CDMDGC » et « CDMDSC » dont le rôle est uniquement de mettre à jour des données. Les messages ayant pour entité « CDMCUS » ou « CDMDUC » se traitent quant à eux très rapidement car il s'agit ici uniquement de mettre à jour des tables personnalisées avec des requêtes SQL simples.

Comme pour chaque entité le traitement des iDocs ne nécessite pas d'attendre la fin de l'ensemble du transfert des messages, la durée totale d'exécution de la migration initiale est d'environ 14 heures. Si le traitement en séquence stricte avait été conservé alors la durée aurait été de plus de six jours.

5.5.2 Migration pour chaque site

Lors de l'implémentation d'un site de vente sur le système SAP, la migration des clients diffère selon qu'ils sont inconnus du système ou déjà présents car utilisés dans d'autres sites actifs. Les clients déjà présents sont simplement étendus d'une nouvelle extension de vente, tandis que ceux qui n'existent pas seront entièrement créés. A ce stade, l'interface d'extraction des clients « CustomerMaster » est exécutée puis CDM procède pour les acheteurs déjà connus à une classification automatique dans la hiérarchie, déclenchant donc systématiquement des transferts de messages « CDMReplicator » avec pour entité « CDMRCM ». En revanche, l'opération de

classification automatique ne peut pas être exécutée pour les clients entièrement recréés dans SAP. Ces derniers disposent en effet logiquement d'une numérotation différente par rapport à l'ancien progiciel de gestion intégré, ce qui ne permet pas d'appliquer la logique de classification automatique. Pour combler ce problème, les administrateurs CDM ont conçu un programme spécifique qui une fois alimenté d'un fichier de correspondance entre les anciens et les nouveaux codes, procède à la renumérotation complète des clients concernés. Une fois exécuté, les modifications réalisées par ce programme dans la hiérarchie sont automatiquement répliquées dans SAP.

L'exécution de ce processus pour chaque site requiert un travail de coordination complexe. Le coordinateur doit ainsi organiser le travail entre l'équipe de migration SAP, les administrateurs CDM, les utilisateurs clefs pour la validation des données, les analystes système de l'équipe Order to Cash (OTC) pour la gestion des cas spéciaux et le spécialiste des interfaces.

Toute migration de données dans SAP étant susceptible de déclencher automatiquement des interfaces d'extraction, nous avons conçu un mécanisme d'exclusion basé sur l'analyse des identifiants des utilisateurs. Ainsi grâce à ce contrôle réalisé au moment de l'analyse des pointeurs de modification, tout changement de données enregistré dans le système par un utilisateur dont l'identifiant débute par « MIG_ » ne déclenchera pas d'interface. Le pointeur sera par ailleurs automatiquement marqué comme traité afin d'éviter son analyse inutile lors de chaque déclenchement du programme de traitement.

L'usage d'un tel procédé ajoute de la flexibilité dans la coordination de la migration d'un site. En travaillant toujours avec des identifiants débutant par « MIG_ », l'équipe de migration SAP n'a pas à se soucier de l'état de préparation des administrateurs CDM car l'interface « CustomerMaster » ne sera jamais déclenchée automatiquement du fait de leurs actions. Une fois que les administrateurs CDM seront prêts, le coordinateur demandera au spécialiste des interfaces de déclencher le transfert des clients modifiés vers CDM par le biais de l'interface d'extraction « CustomerMaster ». Pour se faire, il suffira d'exécuter la transaction BD12 avec en entrée la liste des numéros des clients et le type de message « ZGLDEBMAS ».

La durée d'exécution de ce processus de migration varie selon le nombre de clients utilisés dans le site à implémenter sur SAP. Néanmoins, bien que le temps de traitement effectif dépasse rarement une demi-journée, les contraintes humaines et les vérifications des utilisateurs requièrent à chaque itération de prévoir deux à trois jours entiers.

5.6 Exploitation

5.6.1 Exemple de flux

Lorsqu'un nouveau client est créé sur SAP avec une extension de vente, des pointeurs sont automatiquement générés. L'interface d'extraction se déclenche alors, puis le BDM « CustomerMaster » est envoyé vers CDM. Une fois le nouveau client classifié, l'interface « CDMReplicator » transfère la hiérarchie de ce dernier vers SAP. Le détail des messages

envoyés est listé ci-dessous. Cet exemple illustre le cas d'un nouveau client pour lequel une nouvelle hiérarchie sera crée dans CDM.

1. Création dans SAP du nouveau client « Exemple SA » de type tiers.
2. L'interface « CustomerMaster » est déclenchée mais le message est filtré dans WebMethods car le client ne dispose pas d'extension de vente.
3. Création d'une extension de vente pour le site Argenteuil sur le client « Exemple SA ».
4. L'interface « CustomerMaster » est déclenchée et les données sont envoyées à CDM. Le client n'est pour lors pas classifié.
5. Mise en place de la hiérarchie du client dans CDM. Les niveaux groupe, sous groupe et unique sont créés par les utilisateurs. L'interface « CDMReplicator » est déclenchée pour le transfert dans l'ordre des entités suivantes:
   - Groupe (CDMGRP)
   - Sous groupe (CDMSUB)
   - Unique (CDMCUS)
6. Assignation dans CDM du client « Exemple SA » à la nouvelle hiérarchie.
7. L'interface « CDMReplicator » est déclenchée pour le transfert de l'entité d'affectation du client tiers (CDMRCM)
8. Le traitement des messages met à jour dans SAP la hiérarchie du client « Exemple SA ».

5.6.2 Stabilisation

La complexité de l'interface de chargement dans SAP des hiérarchies a nécessité une longue phase de stabilisation. Par exemple, la gestion du mécanisme de liaison entre les clients de type tiers et les sous groupes au travers du niveau caché de client unique était alors imparfaite. En conséquence, durant cette période les équipes de support devaient régulièrement modifier manuellement des entrées dans la table ZGL_OTC_CDMUNSUB afin de pouvoir débloquer la file d'attente de traitement des iDocs. Plus tard lorsque la charge de travail sur la stabilisation des autres processus faiblit, une refonte de la partie technique de l'interface fut engagée. Quant à elle, l'interface d'extraction des clients a dû subir deux adaptations. La première afin de ne pas interrompre entièrement le transfert d'un client lorsque la translation des codes est invalide que sur l'une de ses extensions de vente, puis une seconde afin de permettre de gérer les effacements de clients que sur l'un des sites de vente.

Après plusieurs mois d'exploitation sur cette nouvelle version, ces deux interfaces sont dorénavant arrivées à maturité. Les seules erreurs survenant encore de manière rares sont liées à des problèmes de qualité de données qu'il suffit pour la plupart de régler en demandant aux utilisateurs de corriger les valeurs erronées dans SAP ou dans CDM.

5.6.3 Statistiques

Chaque jour, une dizaine de nouveaux clients sont créés sur le système SAP. Le processus de préparation des données sur les acheteurs implique des actions de la part d'utilisateurs issus de

différentes familles de rôles. Ainsi avant de pouvoir être utilisé pour la première fois dans une commande, un objet client SAP subit en moyenne cinq modifications. Ajouté aux changements d'usages tels que les modifications de numéros de téléphone, l'interface « CustomerMaster » déclenche quotidiennement près de 200 iDocs.

Le BDM « CDMReplicator » est quant à lui publié en moyenne 400 fois par jour. Ces déclenchements sont influencés par la création de nouveaux clients mais aussi par les changements de structures ou les rachats entre des groupes. De plus, le mécanisme de publication par entité a pour effet d'engendrer l'envoi de plusieurs messages pour une seule action réalisée dans CDM.

Lors des phases de démarrage des nouveaux sites, ces moyennes sont largement surélevées par les échanges liés à la migration de données et les fréquents besoins d'ajustement des valeurs dans le système SAP.

5.6.4 Avenir

L'avenir de l'application CDM chez Givaudan est aujourd'hui incertain. Hébergé sur un système de type IBM AS400 dont la société souhaite se débarrasser dans les prochaines années, une option d'alternative est de permettre aux utilisateurs de maintenir les hiérarchies directement dans le progiciel SAP ECC. Ainsi, pour ne pas perturber les autres systèmes souscrivant au BDM « CDMReplicator », l'idée serait de publier ce dernier à partir de SAP. Néanmoins pour permettre de progressivement s'affranchir du niveau de hiérarchie unique, un pont pourrait être conçu entre cet ancien BDM et un nouveau format visant à progressivement remplacer ce premier. Cette option est schématisée ci-dessous.

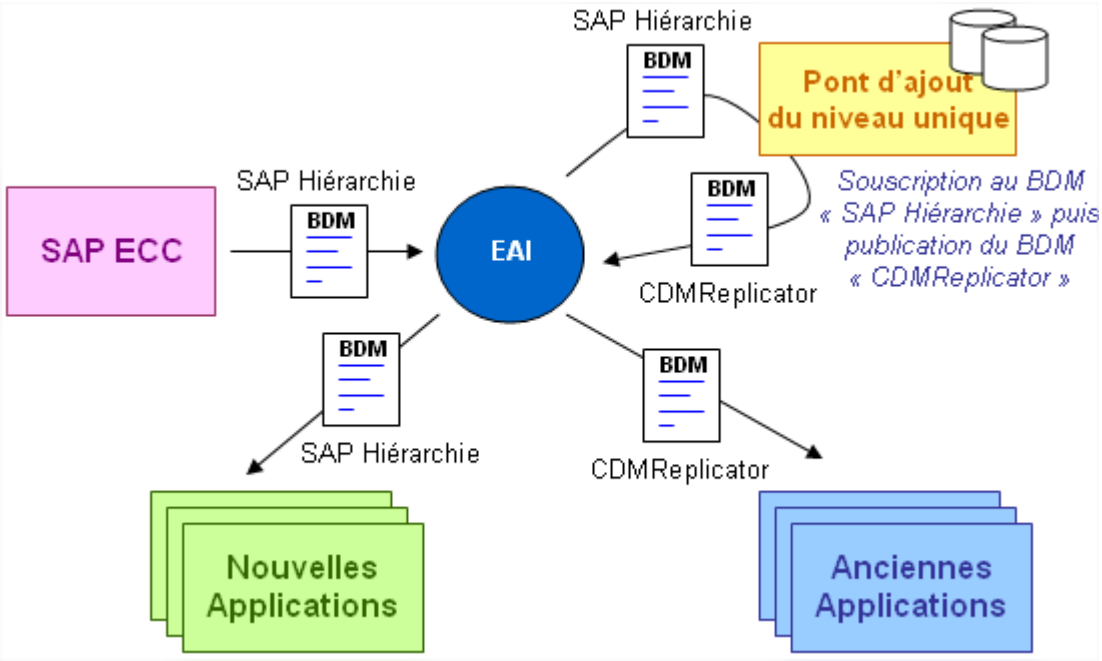


Illustration 44: Prototype de pont entre un BDM « SAP Hiérarchie » et le BDM « CDMReplicator ».

6 Cas 2: Gestion d'un cas de synchronisation complexe avec l'application CMS

A travers le module Production Planning (PP), SAP ECC dispose d'une multitude de solutions permettant de gérer les processus de fabrication des usines. Néanmoins depuis près de vingt ans, Givaudan développe une application nommée CMS, acronyme de « Compounding Management System » afin de gérer les mécanismes de production de manière très optimisée aux contraintes de la société. Grâce à ce système, mais aussi à une multitude d'autres automatismes mis en place dans le progiciel de gestion intégré, Givaudan assure dans la plupart de ses usines un délai de maximum cinq jours entre l'émission d'une commande par un client et le départ du camion de livraison. CMS étant par conséquent une application stratégique à forte valeur ajoutée pour la société, les détails de son fonctionnement du point de vue métier ne seront volontairement pas abordés dans ce document afin ne pas violer les contraintes de confidentialité.

Autrefois connecté au progiciel BPCS, CMS doit être adapté pour s'intégrer avec SAP ECC. Ce changement majeur est l'occasion de revoir en détail la communication entre l'application et le progiciel de gestion intégré. En effet, les limitations de BPCS et les nouveaux concepts métier mis en place au cours du projet Outlook changent le contexte d'intégration de CMS. Ainsi, en parallèle de l'implémentation de SAP ECC, un projet d'évolution de CMS fut initié afin de concevoir une version de l'application pouvant s'interfacer avec le nouveau progiciel et bénéficier des possibilités apportées par ce dernier.

Chaque jour un processus appelé MRP, acronyme de « Material Requirements Planning » est déclenché dans le progiciel de gestion intégré. Connu dans le monde de l'industrie, l'objectif du processus MRP est d'analyser l'ensemble des commandes ouvertes et des stocks de matière premières et de produits finis enregistrés dans le système afin de créer automatiquement des ordres de fabrication qui devront être traités par l'usine. Dans le système informatique mis en place chez Givaudan, les ordres de fabrication sont alors automatiquement transférés par le biais d'interfaces vers l'application CMS. En fonction d'une multitude de paramètres, CMS détermine les unités de production qui seront chargées de traiter l'ordre, puis coordonne l'exécution de l'ensemble des tâches nécessaires à la fabrication des produits finis. Pour ce faire, CMS doit disposer de nombreuses informations sur les produits et leurs formules ainsi que les identifiants et la position des unités de stock. De plus, d'une manière continue au cours d'un processus de fabrication, CMS communique avec SAP pour mettre à jour les états des ordres, puis procéder aux enregistrements des consommations de stocks de matières premières et des entrées des produits finis.

Du point de vue de l'atelier, CMS communique avec les robots exécutant des opérations de fabrication automatisées, puis avec les agents chargés de procéder à des traitements manuels tels que des pesées, des déplacements de stocks, la mise en place des étiquettes sur les emballages ou encore les changements de récipients et d'outils de conception.

Les prochains paragraphes n'aborderont pas en détail la conception de chacune des interfaces entre SAP ECC et l'application CMS. Néanmoins, leur aspect fonctionnel sera brièvement décrit puis les contraintes spécifiques de synchronisation seront présentées. Les solutions à ces problèmes seront alors détaillées et analysées en condition d'exploitation.

6.1 Architecture

Les échanges entre SAP et l'application de gestion de la production se font au travers de dix interfaces également réparties entre les flux sortants et entrants. Conçu comme un système multi-instances, CMS nécessite d'être synchronisée de manière très rapide avec le progiciel de gestion intégré. Ainsi, toutes ses interfaces ont été conçues pour fonctionner au travers de l'intergiciel EAI WebMethods. L'illustration ci-après représente l'architecture générale de la solution ainsi que la totalité des flux entre les deux applications.

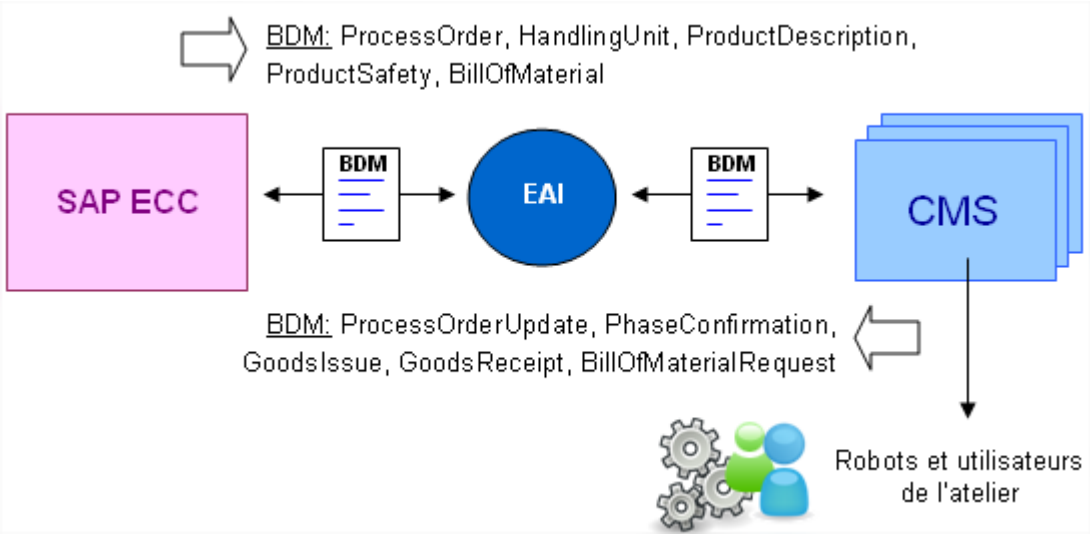


Illustration 45: Flux de communications de l'application CMS.

Ci-dessous la description de chaque BDM échangeant des données de SAP ECC vers CMS, ou inversement.

ProcessOrder (SAP à CMS)

Le BDM « ProcessOrder » a pour rôle de transférer les données sur les ordres de fabrication de SAP vers CMS. Cette interface se déclenche lorsqu'un nouvel ordre est créé par le processus MRP ou lorsque des changements sont opérés dans SAP. Néanmoins, toutes les modifications sur les données des ordres de fabrication ne provoquent pas un déclenchement de l'interface.

ProcessOrderUpdate (CMS à SAP)

L'interface « ProcessOrderUpdate » permet à CMS de mettre à jour dans SAP les données sur les ordres de fabrication. Ceci comporte principalement des états et d'autres informations nécessaires à la supervision de l'activité de fabrication à partir de SAP. Cette interface peut par conséquent être déclenchée plusieurs fois au cours du traitement d'un ordre.

PhaseConfirmation (CMS à SAP)

Les tâches de chaque ordre étant à l'image de l'exécution d'une recette de cuisine divisées en phases, le BDM « PhaseConfirmation » permet à CMS de confirmer la fin du traitement de chacune. Il est généralement déclenché plusieurs fois pour chaque ordre, en fonction du nombre de phases contenues dans ces derniers.

GoodsIssue (CMS à SAP)

Chaque consommation de matière première entrainant la réduction des quantités disponibles dans les unités de stock, le rôle du BDM « GoodsIssue » est de permettre à CMS d'enregistrer automatiquement ces consommations dans SAP. Cette interface se déclenche par conséquent une multitude de fois pour chaque ordre de fabrication.

GoodsReceipt (CMS à SAP)

A la fin de chaque fabrication, l'entrée en stock du produit fini est enregistrée au travers du BDM « GoodsReceipt ». Cette interface est déclenchée uniquement une seule fois par ordre de fabrication.

BillOfMaterialRequest (CMS à SAP)

L'interface « BillOfMaterialRequest » est déclenchée lorsque pour le traitement d'un ordre de type échantillon, CMS nécessite de connaitre la formule de conception du nouveau produit. SAP transmet alors en réponse la formule au travers du BDM « BillOfMaterial ».

BillOfMaterial (SAP à CMS)

Formule de conception d'un produit transférée de SAP vers CMS en réponse à une demande reçue au travers du BDM « BillOfMaterialRequest ».

HandlingUnit (SAP à CMS)

Le BDM « HandlingUnit » est envoyé de SAP vers CMS pour chaque unité de stock créée, modifiée ou supprimée. Le message comporte un numéro identifiant l'unité, les coordonnées de sa localisation physique et de l'entrepôt, le nom du produit contenu et le volume restant disponible dans l'unité. Plus de 200 000 messages « HandlingUnit » sont envoyés chaque jour au travers d'un mécanisme d'analyse déclenché chaque minute. Ce volume et cette fréquence en fait l'interface la plus exigeante du système informatique de Givaudan.

ProductDescription (SAP à CMS)

L'interface « ProductDescription » transfère de SAP vers CMS les données principales sur les produits parmi lesquelles le code, le nom et l'existence dans chaque site. Initialement conçue lors du projet Outlook pour les besoins de l'application CMS, cette interface est aujourd'hui réutilisée par huit applications. Elle est déclenchée à chaque fois qu'un produit est créé ou que ses données sont modifiées dans le système.

ProductSafety (SAP à CMS)

Le BDM « ProductSafety » transfère des informations en complément de l'interface « ProductDescription ». Stockées dans un module spécifique de SAP nommé EH&S, acronyme de « Environment, Health & Safety », les données contenues dans le BDM « ProductSafety » représentent des caractéristiques physiques telles que la densité, le poids, la couleur, le point d'éclair et d'autres informations indiquant de la dangerosité d'un produit. Le module EH&S est alimenté par des analyses faites par des laboratoires de Givaudan et des moteurs de calcul de caractéristiques physiques extrêmement complexes mis en place par l'équipe « Compliance » du projet Outlook..

6.2 Contraintes de synchronisation

Telle que pour ces interfaces entre le progiciel de gestion intégré SAP ECC et l'application CMS, la mise en place d'une multitude d'échanges entre deux systèmes requiert généralement de veiller attentivement aux contraintes de synchronisation. En effet, le dialogue établi entre des applications ne peut aboutir à la réalisation de processus métiers corrects que si les messages sont traités dans le bon ordre.

Les contraintes de synchronisation des transferts allant de SAP vers CMS sont simples. Une à une, chacune des interfaces doit publier les BDM dans le même ordre que les événements métiers sont réalisés dans SAP. Ainsi, si deux modifications de données sont exécutées par des traitements sur la même unité de stock, alors le message publiant les informations du premier changement devra être envoyé et traité avant celui du second changement. La mise en place d'un tel mécanisme est standard tant pour SAP que pour l'intergiciel WebMethods. Ainsi, aucun développement spécifique n'est requis. Pour éviter tout blocage général, une erreur lors du traitement d'un message ne doit en aucun cas interrompre l'exécution des transactions suivantes. Une fois arrivés dans CMS, les messages seront traités selon leur ordre de réception enregistré par un compteur automatique.

Une particularité existe pour les interfaces « ProcessOrder » et « ProcessOrderUpdate ». Agissant toutes deux sur les ordres de fabrication de SAP, il est possible que lorsque CMS transfère une mise à jour de l'ordre via le BDM « ProcessOrderUpdate », cette dernière ne soit plus applicable en raison d'un état de l'ordre dans SAP différent de celui connu par CMS. Ce cas s'explique par le fait que ces deux interfaces sont asynchrones. Ainsi, dans le cas où lorsqu'un message est envoyé une transaction de l'autre système est en attente de traitement, alors il est probable qu'une erreur de synchronisation se produise.

Le traitement par SAP des messages reçus depuis CMS est quant à lui beaucoup plus complexe. Pour chaque ordre de fabrication, les transactions envoyées par CMS doivent être traitées dans leur exacte séquence de publication, indépendamment du type de chacune. Ainsi, si les formats de BDM envoyés pour un même ordre de fabrication sont publiés dans la séquence « Goods

Issue », « PhaseConfirmation » et à nouveau « GoodsIssue », alors cet ordre devra être absolument conservé pour le traitement des iDocs dans SAP. Malheureusement, aucun mécanisme standard de SAP ne permet de réaliser ce type de synchronisation. Le programme de traitement des iDocs entrants « RBDAPP01 » regroupe tous les messages par type avant de procéder à leur exécution.

Un second problème de synchronisation subsiste sur les messages reçus par SAP. En effet, la majorité des traitements engendrent la mise en place d'un verrou soit sur les objets ordre de fabrication, soit sur les unités de stock. En conséquence lorsque le verrou est positionné, tout autre traitement souhaitant accéder en mode modification au même objet échouera.
Par ailleurs pour certains types de messages, une fois l'iDoc exécuté des traitements en arrière plan subsistent et conservent les verrous sur les objets pendant une période de quelques secondes, déterminée en fonction du chargement du système. Ainsi, si le traitement d'un autre iDoc impactant le même objet est déclenché au cours de cette période, alors son exécution terminera immédiatement avec un message d'erreur. Le tableau ci-dessous liste les interfaces CMS entrantes dans SAP avec pour chacune les types d'objets bloqués et l'éventuelle durée des traitements subsistants en arrière plan après l'exécution des iDocs.

BDM	Type de message de l'iDoc	Objet bloqués	Verrou subsistant après le traitement
ProcessOrderUpdate	ZGLUPPO	Ordre de fabrication	Oui, environ 30 secondes
PhaseConfirmation	ZGLTIMETICKET	Ordre de fabrication	Non
GoodsIssue	ZGLGOODSISSUEHU	Ordre de fabrication et unité de stock	Non
GoodsReceipt	WMMBXY	Ordre de fabrication et unité de stock	Oui, environ 45 secondes
BillOfMaterialRequest	ZGLBOMREQ	Aucun	Non

Tableau IV: Caractéristiques de blocage des interfaces entrantes dans SAP depuis CMS.

Si le programme standard de déclenchement du traitement des iDocs est utilisé alors non seulement l'ordre d'exécution des messages ne sera pas respecté, mais de plus les contraintes sur les verrous ne seront pas prises en compte. En conséquence de ces deux problèmes, l'exécution sans encombre d'un processus de fabrication sera conditionnée à un espacement de plusieurs minutes entre l'envoi de chaque message par CMS ! La mise en place d'un mécanisme de retardateur dans CMS n'étant pas une option acceptable, une solution doit être conçue pour permettre le traitement des messages le plus rapidement possible en respectant les contraintes de synchronisation. Cette solution ne devra pas, comme pour les interfaces d'extraction, interrompre la totalité des traitements en cas d'erreur au cours de l'exécution d'un seul message.

Pour permettre d'exécuter très rapidement les messages, chaque site CMS devra disposer dans SAP d'un programme de traitement ordonnancé dédié. Chacun prendra en charge en exclusivité la

responsabilité de l'exécution des iDocs de son site. Les quelques usines où l'activité de production est plus intense que les autres devront quant à elles pouvoir disposer d'une instance du programme de traitement pour chaque atelier géré dans le site CMS. Ces programmes devront être ordonnancés pour être déclenchés chaque minute.

Un traitement rapide des messages est vital pour limiter les incidents au cours de l'exécution du processus métier. Par exemple, tant que l'ordre de fabrication n'est pas disposé dans l'état terminé les utilisateurs sont dans l'incapacité de procéder à l'entrée en stock du produit fini. De plus, la durée de latence entre l'exécution physique d'une consommation de matière première et son enregistrement dans le progiciel SAP ECC est une période au cours de laquelle le système peut réclamer une nouvelle consommation alors que la quantité restante dans l'unité de stock n'est pas suffisante.

6.3 Solutions

6.3.1 Version des ordres de fabrication

Une solution simple fut implémentée pour régler le problème de synchronisation entre les messages « ProcessOrder » et « ProcessOrderUpdate ». Pour chaque ordre de fabrication, une nouvelle zone contenant un numéro de version fut créée dans SAP et ajoutée au contenu du BDM « ProcessOrder ». Initialement affectés avec le numéro de version « 1 », chaque modification des ordres réalisée au travers d'une mise à jour faite par un utilisateur ou par l'exécution d'un message « ProcessOrderUpdate » déclenche un incrément de ce numéro. Lorsqu'au travers de l'interface « ProcessOrderUpdate » CMS transmet au progiciel de gestion intégré une demande de mise à jour des informations de l'ordre, le numéro de version sur lequel CMS s'est basée pour déterminer ce besoin est envoyé. Au cours du traitement de l'iDoc, SAP compare le numéro de version reçu avec celui enregistré dans le système et annule le message au cas où ce premier est plus ancien. Enfin, lorsque CMS réceptionne un message « ProcessOrder » avec un nouveau numéro de version, celui ci détermine en fonction des informations reçues s'il est nécessaire de republier ou non un message « ProcessOrderUpdate ».

6.3.2 Séquencement du traitement des messages entrants dans SAP

Les problèmes sur le mécanisme de traitement dans SAP des messages venus de CMS peut être décomposé en trois parties. Une première solution devra être trouvée pour permettre de faire une distinction des messages en fonction de leur site ou atelier de provenance. Grâce à cette solution il sera alors possible d'ordonnancer des programmes exécutant les messages d'une seule instance. Un second mécanisme devra être mis en place afin de s'assurer que pour chaque site ou atelier les iDocs seront traités dans le bon ordre. Il s'agira ici de casser le mécanisme standard de SAP qui regroupe les messages en fonction de leur type. Enfin, la troisième solution devra ajouter au second mécanisme une logique permettant de prendre en compte les contraintes des verrous au cours de l'exécution des iDocs. Chacune de ces solutions est expliquée ci-après dans un paragraphe dédié.

6.3.2.1 Distinction des messages

Deux solutions complémentaires furent sélectionnées pour permettre la distinction des messages. La première est d'affecter dans SAP un profil de partenaire dédié à chaque usine. Ainsi les iDocs de l'usine d'Argenteuil, en France seront créés dans le profil « CMS_AR01 ». La seconde est d'enregistrer dans un champ dédié des iDocs un code identifiant l'atelier à l'intérieur de l'usine. Le programme de traitement des iDocs devra comporter ces deux champs dans son écran de déclenchement afin de permettre une sélection des messages en fonction de ces critères.

Pour les messages entrants, la détermination du profil de partenaire se fait dans l'intergiciel EAI au moment du mapping de l'iDoc. L'information est enregistrée dans le segment d'entête « EDI_ DC40 » commun à tous les iDocs, quel que soit leur type. Une simple logique concaténant le code de l'usine à « CMS_ » permet de mettre en place cette détermination. Le code de l'atelier sera quant à lui mappé dans une zone inutilisée du segment d'entête. Afin que WebMethods puisse réaliser ce travail, chacun des BDM sortant de CMS à destination de SAP doit être modifié pour contenir le code de l'usine et de l'atelier.

6.3.2.2 Mise en séquence des traitements

Pour permettre la mise en séquence des traitements des iDocs dans SAP, la première étape est de s'assurer que les messages sont envoyés et reçus dans le bon ordre. Une solution simple et utilisée depuis longtemps avec WebMethods fut utilisée. Il s'agit de créer dans CMS une table de déclenchement unique, appelée table de notification globale. Pour chaque BDM à publier, CMS devra écrire dans cette table un enregistrement contenant un code de type de message, un identifiant de transaction et un compteur qui sera incrémenté pour chaque nouvelle insertion. A intervalle régulier, WebMethods sélectionnera les enregistrements de cette table dans l'ordre ascendant du champ compteur, puis déclenchera pour chaque code, type de message et identifiant de transaction la publication du BDM correspondant. A la fin de ce processus, tous les enregistrements de la table de notification globale sont supprimés afin d'éviter que les BDM soient republiés à la prochaine analyse de l'EAI.

Ainsi, car ils seront publiés dans l'ordre voulu par CMS, WebMethods conservera cet ordre au cours du processus de mapping entre les BDM et les iDocs. Afin qu'une fois reçu dans SAP ce dernier puisse toujours être en mesure de déterminer l'ordre des messages, le champ « SERIAL » des segments d'entête « EDI_DC40 » de chaque iDoc doit être renseignée avec une valeur numérique unique qui sera toujours supérieure au dernier iDoc envoyé.

Par défaut, le programme standard SAP de traitement des iDocs « RBDAPP01 » trie les messages en fonction de la valeur contenue dans le champ « SERIAL ». Néanmoins, c'est ensuite que le regroupement par type de message est réalisé. Ainsi, pour s'assurer du traitement en séquence selon le champ « SERIAL » et sans tenir compte du type de message, il suffit de concevoir un

programme de déclenchement identique au programme « RBDAPP01 » dans lequel le mécanisme de regroupement par type sera supprimé.

6.3.2.3 Gestion des verrous

La mise en place de la gestion des verrous consiste à ajouter dans le programme de traitement des iDocs « RBDAPP01 » un algorithme permettant de déterminer si les conditions sont réunies pour exécuter ou non chaque message. Si elles ne le sont pas, alors l'iDoc devra rester dans l'état « 64 : en attente ». Il sera alors réévalué lors de la prochaine exécution du programme de traitement, ordonnancé toutes les minutes.

Les conditions de déclenchement du traitement d'un iDoc provenant de CMS sont les suivantes:

- L'iDoc est dans l'état « 64 : en attente ».
- Aucun traitement n'est en cours sur l'objet bloqué (ordre de fabrication ou unité de stock).
- Pour le même ordre de fabrication, le traitement d'aucun iDoc précédent dans la file d'attente n'a été retardé pour raison de blocage.

Les verrous n'étant pas toujours relâchés à l'issue de l'exécution d'un iDoc, l'idée est d'empêcher pendant une durée déterminée le traitement de tout autre message impactant les mêmes données. Pour ce faire, une table de mise en quarantaine nommée ZGL_CMS_OBJLOCK fut créée. Celle ci contient les champs suivants:

- Type d'objet. Code identifiant le type de l'objet bloqué. « PO » pour ordre de fabrication ou « HU » pour unité de stock.
- Code d'objet. L'identifiant de l'objet bloqué. Code de l'unité de stock ou numéro de l'ordre de fabrication.
- Heure et date d'expiration. Une fois cet instant passé, le blocage est considéré comme obsolète.
- Numéro d'identifiant de l'iDoc. Utile uniquement pour faciliter la supervision.
- Code d'instance. Le programme de traitement des iDocs devant disposer de plusieurs instances en fonction des usines et ateliers, chacune d'elles doit identifier et disposer ses propres blocages en fonction de son identité.

Les iDocs ne relâchant pas les verrous immédiatement après leur exécution sont identifiés grâce à leur type de message disposé dans une nouvelle zone de l'écran de sélection du programme de traitement. Un second champ doit être créé afin de contenir la durée pendant laquelle l'objet bloqué devra rester en quarantaine.

Un nouvel enregistrement est ajouté dans la table ZGL_CMS_OBJLOCK chaque fois qu'un iDoc ayant un type de message listé comme ne relâchant pas immédiatement le verrou d'un objet est traité. L'heure et la date d'expiration sont disposées en ajoutant le délai de quarantaine à l'heure

courante. Chaque fois qu'il est exécuté, le programme de traitement des iDocs débute par une suppression de tous les enregistrements obsolètes de la table de quarantaine.

Enfin, pour fonctionner le programme doit disposer d'informations sur le contenu des transactions. Afin d'éviter le processus coûteux en performances qui consisterait à analyser dans le programme le contenu de chaque message, le champ « ARCKEY » du segment d'entête des iDocs doit contenir les données clefs pour l'exécution de l'algorithme. Ce champ est habituellement renseigné dans WebMethods avec un identifiant de transaction appelé code d'activation, cette information est utile pour tracer le cheminement des messages. Pour ne pas avoir à la supprimer, il suffit simplement d'y concaténer les nouvelles données avec un séparateur. La zone d'entête « ARCKEY » contient ainsi une valeur formatée tel qu'expliqué ci-dessous.

`Code d'activation WebMethods | Type d'objet | Identifiant d'objet`

- Code d'activation WebMethods: identifiant de la transaction dans l'EAI. Information non utilisée par l'algorithme de traitement des iDocs.
- Type d'objet. « HU » pour une unité de stock, « PO » pour un ordre de fabrication.
- Identifiant de l'objet. Selon le type d'objet, cette zone comporte le code de l'unité de stock ou numéro de l'ordre de fabrication.

Avant le traitement de chaque message, l'algorithme vérifie s'il existe dans la table ZGL_CMS_OBJLOCK un enregistrement comportant le même type et identifiant d'objet. Si un tel enregistrement est trouvé, alors cela signifie que l'objet est en quarantaine. L'iDoc ne sera par conséquent pas traité lors de cette exécution du programme. En revanche, si aucun enregistrement n'est trouvé alors le traitement de l'iDoc sera déclenché. De plus, dans le cas où le type de message de l'iDoc traité est identifié dans la liste de ceux posant des verrous persistants après leur exécution, alors un nouvel enregistrement sera créé dans la table ZGL_CMS_OBJLOCK avec les valeurs correspondantes.

6.4 Conception

Cette section couvre uniquement le développement du programme de séquencement du traitement des messages reçus dans SAP. D'une construction très simple, l'algorithme du système de numérotation des versions des ordres de vérification ne sera pas détaillé ici.

Le programme « RBDAPP01 » ne disposant d'aucun user-exit permettant son extension, il a été nécessaire d'en créer une copie afin de pouvoir la modifier. Néanmoins même si une extension avait été possible, il aurait alors été préférable de l'éviter. En effet, étant utilisé pour le traitement des iDocs de la majorité des autres interfaces entrantes dans SAP, réaliser une modification sur la version originale de ce programme aurait été très risqué.

6.4.1 Ecran de sélection

L'écran de sélection du nouveau programme est représenté ci-dessous de manière volontairement réduite (certaines zones provenant de la version standard du programme et inutiles pour cette explication sont masquées afin de ne pas surcharger ce document).

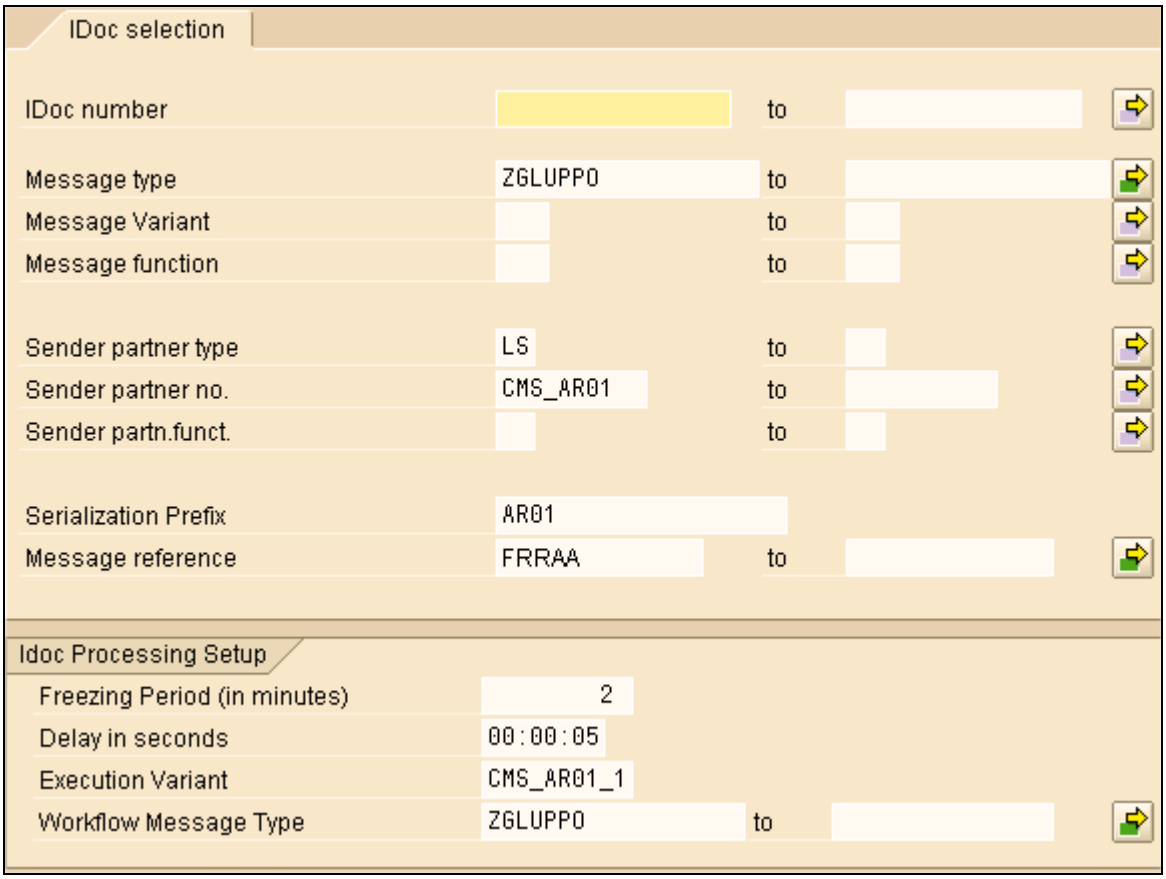


Illustration 46: Ecran de sélection du nouveau programme de traitement des iDocs.

 : Icone permettant de renseigner plusieurs valeurs d'entrée pour un même champ. Le carré vert indique que plusieurs valeurs sont déjà définies. Dans ce cas, seule la première est affichée à l'écran mais toutes seront prises en compte lors de l'exécution du programme.

Les nouvelles zones sont les suivantes:

- « Message reference »: code de l'atelier. Utilisé par le mécanisme de distinction. La zone provenant du segment standard « Sender partner no. » identifie quant à elle le profil de partenaire.
- « Freezing period »: durée de mise en quarantaine. Utilisé par le mécanisme de gestion des verrous.
- « Delay in seconds »: délai avant la prise en compte des nouveaux iDocs. Ajoutée au cours de la période de stabilisation. L'utilité de cette zone sera précisée au cours du chapitre sur l'exploitation.

- « Execution variant »: code d'instance de l'exécution du programme. Cette valeur permet d'identifier de manière unique le traitement ordonnancé qui exécutera le programme pour l'usine et l'atelier correspondant.
- « Workflow message type »: liste des iDocs pour lesquels le traitement ne relâche pas immédiatement le verrou sur les données. Dans le scénario d'intégration avec CMS, les types de message concernés sont ZGLUPPO et WMMBXY correspondant respectivement aux interfaces « ProcessOrder » et « GoodsReceipt ».

6.4.2 Algorithme

L'algorithme du nouveau programme de traitement est détaillé ci-dessous. Ce dernier se déclenche dès lors que l'écran de sélection présenté précédemment est validé. Pour chaque usine ou atelier (selon les cas), un traitement ordonnancé dédié sera configuré pour être déclenché chaque minute avec les paramètres correspondants.

```
Programme  Z_GL_BC_R_CMS_RBDAPP01

* Copié et adapté du programme standard RBDAPP01

* Paramètres d'entrée provenant de l'écran de sélection:
*     variante: champ « Execution variant ».
*     partenaire: champ « Sender partner no. ».
*     code_atelier: champ « Message reference ».
*     Type_message[] : champ « Message type » (plusieurs valeurs).
*     delai_en_minutes : champ « Freezing period ».
*     Verrous[] : champ « Workflow message type » (plusieurs valeurs).


// Effacement des verrous obsolètes
Effacer de la table ZGL_CMS_OBJLOCK tous les enregistrements pour lesquels timestamp < date/heure courante et execution variant = variante (paramètre d'entrée)

// Sélection des iDocs en attente de traitement pour l'usine et de l'atelier
Liste_iDocs[] = Sélectionner tous les iDocs en attente pour lesquels statut = 64 (en attente), partner profile = partenaire, message reference = code_atelier, code message type dans tableau Type_message[].

// Trier l'ensemble des iDocs en fonction de leur numéro de séquence
Trier Liste_iDocs[] en fonction du champ d'entête « SERIAL ».


/* Désactiver  le tri des iDocs en fonction de leur type de message */
// Evaluer les iDocs et déclencher les traitements en fonction des conditions liées de position des verrous.
Boucle Liste_iDocs[] index i
```

```
      // Extraire le type et identifiant d'objet du champ d'entête de l'iDoc
      type_objet = Lire Liste_iDocs[i].ARCKEY, extraire le type d'objet
      code_objet = Lire Liste_iDocs[i].ARCKEY, extraire l'identifiant d'objet

      // Contrôle du verrou
      Vérifier s'il existe un enregistrement dans la table ZGL_CMS_OBJLOCK
      avec execution variant = variante, type d'objet = type_objet et
      identifiant d'objet = code_objet.

            Si oui: ne pas traiter l'iDoc.
            Si non: traiter l'iDoc.
                  // Vérifier si le message traité nécessite la mise en
                     place d'un nouveau verrou.
                  Vérifier si le type de message de l'iDoc traité est listé
                  dans le tableau Verrous[]:

                      Si non: ne rien faire.
                      Si oui: Ajouter un enregistrement dans la table
                      ZGL_CMS_OBJLOCK avec les valeurs suivantes:
                         execution variant = variante,
                         type d'objet = type_objet,
                         identifiant objet = code_objet
                         timestamp = heure courante + delai_en_minutes.

                   Fin vérification

      Fin vérification

Fin boucle

Fin du programme
```

6.5 Exploitation

Le nouveau programme de traitement des iDocs fut développé seulement deux mois avant la mise en production du premier site de Givaudan sur le progiciel SAP ECC. A cette époque, l'instabilité générale de la solution requit de repousser le démarrage d'un mois. Ce nouveau délai fut donc mis à contribution pour trouver et développer cette solution durable aux problèmes de synchronisation.

6.5.1 Stabilisation

Une fois la logique déterminée et l'algorithme spécifié, le développement du programme fut réalisé rapidement. Dès lors, une nouvelle série de tests débuta et ils permirent de confirmer l'efficacité de la solution. Cette première version fut utilisée pour le premier démarrage du projet Outlook.

Quelques mois plus tard, nous nous rendîmes compte que certaines erreurs de synchronisation subsistaient. Aux premiers abords incompréhensibles, une analyse détaillée de ces cas nous fit constater que bien que l'intergiciel WebMethods transfère à SAP ECC les messages dans la séquence désirée, ces derniers peuvent ne pas être enregistrés tous au même instant dans le système. Ainsi, lorsque cinq iDocs sont transférés à SAP, au cours des premiers instants après l'envoi il est possible que seuls les derniers soient enregistrés dans le système. Si à ce moment là le programme de traitement des iDocs se déclenche, alors ces messages seront traités avant les trois premiers iDocs de la série. Pour combler ce problème, un mécanisme de délai de carence fut ajouté au nouveau programme de traitement. Désormais, aucun iDoc reçu dans le système depuis moins de cinq secondes ne sera considéré par le programme. Ces messages seront par conséquent uniquement évalués lors de la prochaine exécution.

Depuis, aucun autre changement ne fut requis sur le programme. Aujourd'hui, grâce à cette solution ce mécanisme traite environ 80 000 messages chaque jour au travers de 12 instances affectées à des usines ou ateliers différents. Chaque démarrage ajoute entre 10 000 et 20 000 messages à ce volume et requiert d'ordonnancer des nouvelles instances.

6.5.2 Exemple

En support d'un exemple simplifié, le tableau ci-dessous liste les messages envoyés par CMS au cours du traitement de deux ordres de fabrication. L'analyse de l'exécution du programme de traitement sera détaillée ensuite.

Heure, minute, seconde	Numéro de séquence des messages	Numéro de l'ordre de fabrication	Interface (BDM)	Verrou
14:30:12	1	10002345	ProcessOrderUpdate	Oui
14:30:13	2,3,4,5,6	10002345	GoodsIssue (6 messages)	Non
14:30:25	7	10006789	ProcessOrderUpdate	Oui
14:31:07	8	10002345	GoodsIssue	Non
14:34:40	9	10002345	ProcessOrderUpdate	Oui
14:34:47	10,11,12,13,14,15	10006789	GoodsIssue	Non
14:35:16	16	10002345	GoodsReceipt	Oui
14:38:22	17	10006789	ProcessOrderUpdate	Oui
14:38:41	18	10006789	GoodsReceipt	Oui

Tableau V: Exemple de flux de messages entrants pour deux ordres de fabrication.

Comme indiqué plus tôt, le programme de traitement se déclenche toutes les minutes. Dans notre exemple nous considérons qu'il se déclenche à ce moment exact et donc sans retard. La mise en place des verrous dans la table ZGL_CMS_OBJLOCK se fait en ajoutant 2 minutes à l'heure d'exécution du programme.

14:30:00 Pas de traitement réalisé car aucun iDoc n'est en attente dans le système.

14:31:00 7 iDocs en attente dans le système.

Le message numéro **1** « ProcessOrderUpdate » pour l'ordre 10002345 est traité. Un verrou est disposé pour cet ordre dans la table ZGL_CMS_OBJLOCK.

Les messages **2**, **3**, **4**, **5** et **6** ne sont pas traités en raison du verrou précédent.

Le message numéro **7** « ProcessOrderUpdate » pour l'ordre 10006789 est traité. Un verrou est disposé pour cet ordre dans la table ZGL_CMS_OBJLOCK.

14:32:00 6 iDocs en attente dans le système, dont 5 non traités à la précédente exécution.

Les verrous étant encore disposés sur les ordres 10002345 et 10006789, aucun traitement n'est réalisé.

14:33:00 6 iDocs en attente dans le système, dont 6 non traités à la précédente exécution.

Les verrous sur les ordres 10002345 et 10006789 sont obsolètes.

Les messages **2**, **3**, **4**, **5**, **6** et **8** sont traités. Aucun verrou n'est disposé car les messages « GoodsIssue » relâchent le blocage des données immédiatement.

14:34:00 Pas de traitement réalisé car aucun iDoc n'est en attente dans le système.

14:35:00 7 iDocs en attente dans le système.

Le message numéro **9** « ProcessOrderUpdate » pour l'ordre 10002345 est traité. Un verrou est disposé pour cet ordre dans la table ZGL_CMS_OBJLOCK.

Les messages **10**, **11**, **12**, **13**, **14** et **15** sont traités. Aucun verrou n'est disposé car les messages « GoodsIssue » relâchent le blocage des données immédiatement.

14:36:00 1 iDoc en attente dans le système.

Le message numéro **16** « GoodsReceipt » pour l'ordre 10002345 n'est pas traité en raison du verrou dans la table ZGL_CMS_OBJLOCK.

14:37:00 1 iDoc en attente dans le système, non traité lors de la précédente exécution.

Le verrou sur l'ordre 10002345 est obsolète.

Le message numéro **16** « GoodsReceipt » pour l'ordre 10002345 est traité. Un verrou est disposé pour cet ordre dans la table ZGL_CMS_OBJLOCK.

14:38:00 Pas de traitement réalisé car aucun iDoc n'est en attente dans le système.

14:39:00 2 iDocs en attente dans le système.

Le verrou sur l'ordre 10002345 est obsolète.

Le message numéro **17** « ProcessOrderUpdate » pour l'ordre 10006789 est traité. Un verrou est disposé pour cet ordre dans la table ZGL_CMS_OBJLOCK.

Le message numéro **18** « GoodsReceipt » pour l'ordre 10006789 n'est traité en raison du verrou précédent.

14:40:00 1 iDoc en attente dans le système.

Le message numéro **18** « GoodsReceipt » pour l'ordre 10006789 n'est pas traité en raison du verrou dans la table ZGL_CMS_OBJLOCK.

14:41:00 1 iDoc en attente dans le système.

Le verrou sur l'ordre 10006789 est obsolète.

Le message numéro **18** « GoodsReceipt » pour l'ordre 10006789 est traité. Un verrou est disposé pour cet ordre dans la table ZGL_CMS_OBJLOCK.

14:42:00 Pas de traitement réalisé car aucun iDoc n'est en attente dans le système.

14:43:00 Pas de traitement réalisé car aucun iDoc n'est en attente dans le système.

Le verrou sur l'ordre 10006789 est obsolète.

A l'issue de ce cycle, tous les iDocs des deux ordres de fabrication sont traités. Transférés sur une période de 8 minutes, le traitement complet aura été réalisé en dix minutes (la première et les deux dernières itérations ne sont pas considérés dans ce calcul car aucun iDoc non traité du cycle n'était présent dans le système à ces instants).

7 Cas 3: Intégration des données de planification avec l'application ORTEMS

ORTEMS est une application utilisée dans de nombreuses entreprises du domaine de l'industrie. Elle est éditée par une société portant le même nom que le produit. L'objectif d'ORTEMS est de réaliser des plans de planification d'opérations de production en fonction d'une multitude de contraintes. Contrairement au système CMS, Givaudan utilise ORTEMS pour planifier des tâches sur le long terme. Par ailleurs, les types de fabrications gérées sont différents. Les ordres planifiés au travers d'ORTEMS sont essentiellement créés pour concevoir des composants intermédiaires, c'est à dire des produits semi-finis qui seront utilisés dans d'autres ordres de fabrication. Ces tâches étant ainsi particulières et s'exécutant pour la plus part de manière manuelle, aucune fabrication planifiée par ORTEMS ne verra sa conception être coordonnée par l'application CMS.

Bien que SAP propose une solution équivalente au travers du module APO, acronyme de « Advanced Planning Organizer », il fut décidé d'utiliser ORTEMS pour des contraintes de coût et de complexité. L'application est par ailleurs déjà présente dans le système informatique de Givaudan et son usage ne concerne qu'un nombre très limité de sites de production.

Pour fonctionner, ORTEMS doit disposer d'une multitude d'informations en provenance du progiciel de gestion intégré: prévisions de demandes, ordres de fabrication, spécification des produits, formules, recettes, outils de fabrication à disposition dans l'usine et état des stocks pour les produits gérés. Chaque jour, un traitement est déclenché afin de réévaluer toutes les données reçues du progiciel et de procéder à un nouveau calcul complet du planning. Une fois cette tâche terminée, les dates et horaires d'exécution des travaux sont renvoyés à SAP au travers d'interfaces.

La solution conçue doit pouvoir prendre en compte des règles de calcul de planning différentes en fonctions des sites de production sur lesquels elle est utilisée. Ainsi, il s'agit de concevoir une architecture assez globalisée pour réduire les coûts de développement et faciliter la maintenance, tout en étant assez flexible pour pouvoir prendre en considération les contraintes locales de chaque site.

A travers ce chapitre, l'architecture d'intégration entre le système SAP ECC et l'application ORTEMS est présentée. Les intergiciels WebMethods (EAI) et DataStage (ETL) étant tout deux utilisés, seules les interfaces conçues sur ce dernier seront détaillées. Le développement des interfaces utilisant l'intergiciel WebMethods est quant à lui similaire aux exemples abordés au cours des précédents chapitres.

7.1 Architecture

Le calcul du planning ne s'exécutant qu'une seule fois par jour, il est inutile de synchroniser ORTEMS de manière régulière avec des interfaces de type EAI. Néanmoins certaines extractions étant déjà disponibles sur WebMethods au travers de flux construits pour d'autres systèmes, il est dans ce cas plus simple et moins coûteux de bénéficier de l'existant plutôt que de construire des nouvelles interfaces de type batch. Les données pour lesquelles aucun flux n'est déjà disponible seront quant à elles transférées par le biais d'interfaces réalisées sur l'intergiciel DataStage.

L'application ORTEMS est vendue avec un outil d'intégration appelé VIC, acronyme de « Visual Interface Configurator ». VIC permet de se connecter au travers du protocole ODBC à la base de données de n'importe quelle application afin d'extraire et de charger des informations. C'est dans cet outil que réside la logique de mapping des données. Bien qu'il existe un additif pour permettre à VIC de se connecter avec des systèmes de type SAP, il fut décidé de ne pas l'utiliser afin de ne pas multiplier les applications se connectant de manière directe à SAP ECC. L'usage des intergiciels WebMethods et DataStage pour la plupart des interfaces est privilégié afin de faciliter le support, la supervision et les évolutions. Ainsi, il fut décidé de concevoir une nouvelle base de données à l'extérieur de SAP dans laquelle toutes les informations nécessaires à ORTEMS seront synchronisées depuis le progiciel de gestion intégré au moyen d'interfaces conçues avec les intergiciels standard. VIC se connectera ainsi à cette source de données et non directement à SAP.

Cette architecture dispose de plus d'un avantage conséquent pour la gestion des contraintes spécifiques dans le cadre d'une solution globale. En effet les interfaces entre SAP et la base de données intermédiaire seront globales, chaque site ORTEMS disposera par conséquent du même format pour les données stockées à cet endroit. En revanche, les règles d'intégration de VIC pourront être différentes dans chaque site. Si dans l'avenir, un nouveau champ est requis dans la base de données intermédiaire pour une instance d'ORTEMS spécifique, alors ce dernier sera ajouté dans toutes les bases. Néanmoins, la règle d'utilisation de la nouvelle zone ne sera mise en place que dans l'instance VIC du site concerné.

Le schéma représenté ci-après illustre l'architecture des interfaces entre SAP ECC et ORTEMS aux travers des outils VIC, WebMethods et DataStage. Les données modifiées sont transférées de manière immédiate par l'EAI WebMethods. En revanche, les actions de VIC et de DataStage sont contrôlées par l'ordonnanceur ControlM. L'enchaînement des tâches pour les scénarios d'importation et d'exportation sera détaillé dans le paragraphe dédié à la conception.

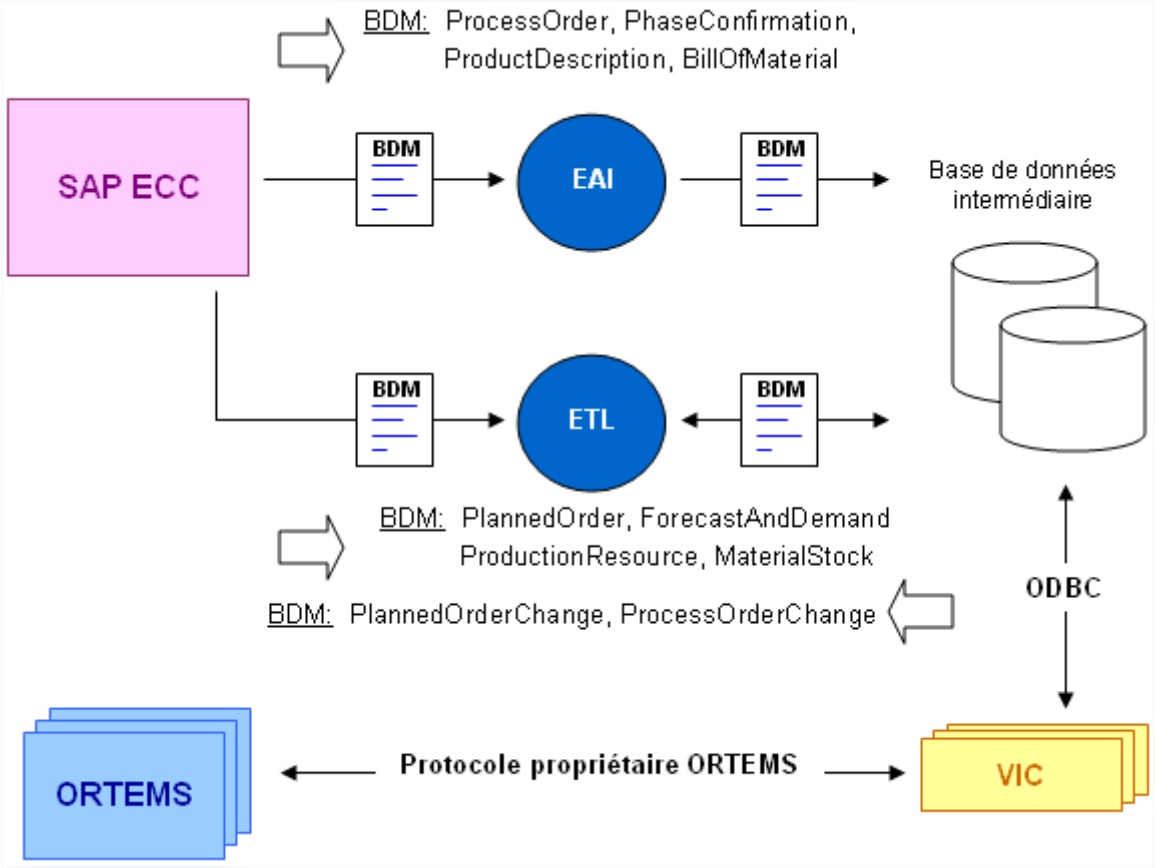


Illustration 47: Architecture des interfaces entre SAP ECC et ORTEMS.

Les interfaces à construire sur l'intergiciel DataStage sont les suivantes:

PlannedOrder (SAP à ORTEMS)

Le BDM « PlannedOrder » représente des ordres de fabrication planifiés sur le long terme. Lorsque le moment de la conception est venu, les ordres planifiés sont transformés en ordres de fabrication.

ForecastAndDemand (SAP à ORTEMS)

A travers ce BDM, SAP envoi à ORTEMS des prévisions quant aux besoins de production. Ces prévisions sont basées sur des commandes ouvertes dans le système et des statistiques calculées sur les données des périodes précédentes.

ProductionResource (SAP à ORTEMS)

Informations sur les outils de production. Etat (actif, panne, maintenance), capacité, temps d'indisponibilité entre deux usages...

MaterialStock (SAP à ORTEMS)

Pour chaque produit dont la production est gérée par ORTEMS ainsi que pour les matières premières utiles à sa conception, le BDM « MaterialStock » transfère la quantité disponible en stock.

PlannedOrderChange (ORTEMS à SAP)

Le BDM « PlannedOrderChange » transfère à SAP les dates d'exécution des ordres planifiés calculées par ORTEMS. C'est à ces dates que les ordres planifiés seront transformés en ordres de fabrication.

ProcessOrderChange (ORTEMS à SAP)

Le BDM « ProcessOrderChange » transfère à SAP des dates, heures d'exécution et ressources de production à utiliser pour l'exécution de chaque tâche des ordres de fabrication.

7.2 Conception

La réalisation d'interfaces de type batch avec SAP ECC au travers de l'ETL DataStage est simple. La technique diffère sensiblement selon que les flux soient sortants ou entrants.

7.2.1 Flux de SAP ECC vers ORTEMS

La mise en place d'une connexion directe entre DataStage et la base de données de SAP étant interdite pour des raisons de support, l'extraction des données depuis SAP se fait au moyen de fichiers texte. La conception des flux allant de SAP vers ORTEMS requiert donc de procéder aux tâches suivantes:

- Création d'un programme d'extraction dans SAP. Le programme doit sélectionner les données dans les tables du système, puis les extraire dans des fichiers texte de type CSV.
- Création des tables dans la base données intermédiaire. Par mesure de simplicité, ces tables doivent être créées avec un format identique à celui des BDM. Ainsi, les développements DataStage devront procéder à des mappings très simplifiés.
- Développement des mappings dans l'intergiciel DataStage.
- Conception de la logique d'intégration dans l'outil VIC.
- Mise en place du traitement ordonnancé pour coordonner l'exécution de l'ensemble des tâches sur les différents types de systèmes.

La conception de la logique d'intégration dans l'outil VIC ne sera pas abordée au cours de cette partie. Ce travail fut réalisé par des consultants ORTEMS et une équipe locale dédiée.

7.2.1.1 Programme d'extraction SAP

Le développement des programmes d'extraction de données dans des fichiers de type texte est simple. En fonction des champs contenus dans le BDM, il suffit d'écrire des requêtes au format ABAP SQL (un dérivé proche du SQL) puis de transférer les résultats dans des fichiers au travers de fonctions standards de SAP. Si la structure des données est plate, c'est à dire dans le cas où une seule requête est nécessaire, alors uniquement un fichier devra être créé. En revanche, si la structure est plus complexe alors plusieurs fichiers d'extraction seront nécessaires.

Les écrans de sélection des programmes d'extraction de chaque interface doivent contenir les informations nécessaires permettant de filtrer les données. Il est de plus nécessaire de renseigner le nom des fichiers ainsi qu'un caractère de séparation qui sera utilisé pour distinguer les valeurs. La plupart des interfaces utilisent la barre verticale « | » comme séparateur. L'illustration ci-dessous représente l'écran de sélection du programme d'extraction des ordres de planification. Les champs « Plants », « MRP Group » et « Prod. Sched. Profile » permettent de filtrer les données à extraire, tandis que « File Name » et « Separator » sont des paramètres de spécification du fichier résultat.

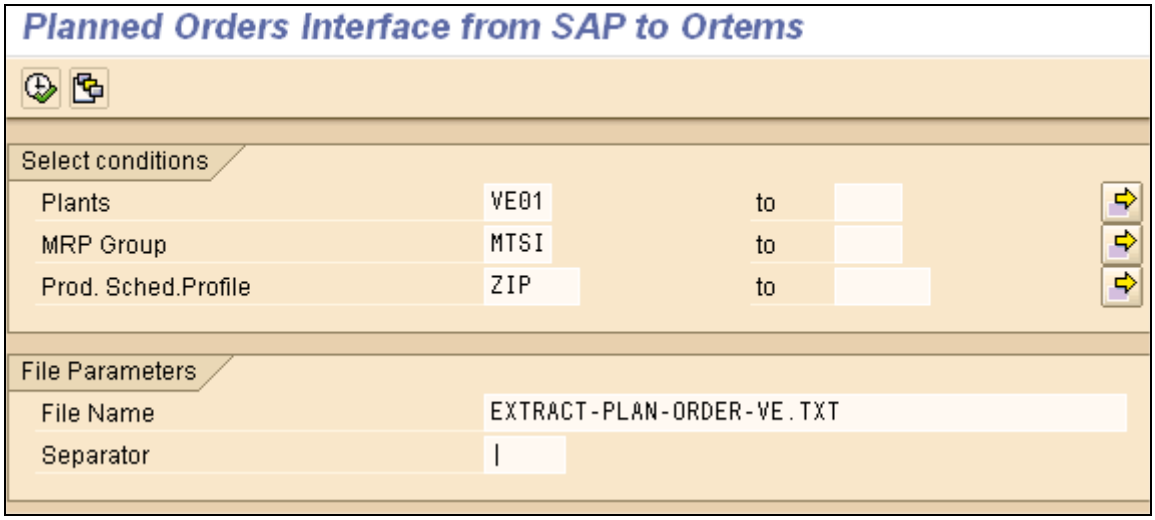


Illustration 48: Ecran de sélection d'un programme SAP d'extraction de données pour DataStage.

Les interfaces de type batch ne font aucune distinction entre les données nouvelles et celles déjà transmises par le passé. Ainsi lors de chaque exécution toutes les informations disponibles dans le système sont sélectionnées, extraites puis traitées sans se limiter aux seuls objets changés.

7.2.1.2 Création des tables dans la base de données intermédiaire

Le SGBD utilisé étant Oracle, la création des tables dans la base de données intermédiaire fut réalisée au travers de l'outil Toad. Cette tâche consista uniquement à créer des tables de structure identique aux formats des BDM, et par conséquent aussi similaires aux fichiers extraits depuis SAP. Une fois terminé, le mapping doit être renseigné dans les onglets « ORTEMS Sub » des documents de spécification des BDM.

7.2.1.3 Développement DataStage

Dans notre scénario d'intégration, les développements sur l'intergiciel DataStage consistent à lire un fichier texte contenant des valeurs séparées par un caractère déterminé, puis à enregistrer chaque ligne du fichier dans une table de même format. Pour éviter les collisions, la première étape sera de vider entièrement les tables de la base de données intermédiaire avant d'insérer les enregistrements issus des fichiers.

DataStage étant optimisé pour l'exécution de processus d'échange de données massifs, les développements se réalisent de manière différente que dans l'EAI WebMethods qui est lui conçu pour les traitements objet par objet. Ainsi, les développements sont découpés en services contenant chacun des étapes appelées « Stages ». Pour chacune, l'exécution du traitement s'effectue le plus rapidement possible et débute dès qu'un premier enregistrement est reçu. Les étapes ne sont donc pas exécutées une à une pour chaque fichier à traiter. Par exemple l'exécution du second « Stage » débute dès que le premier enregistrement est extrait de l'étape initiale. L'illustration ci-dessous représente le développement d'un service DataStage composé d'un service de lecture de fichier texte envoyant des données à un « Stage » de mapping, lui même connecté à un dernier « Stage » insérant les enregistrements dans la table INB_PL_HEADER de la base de données intermédiaire.

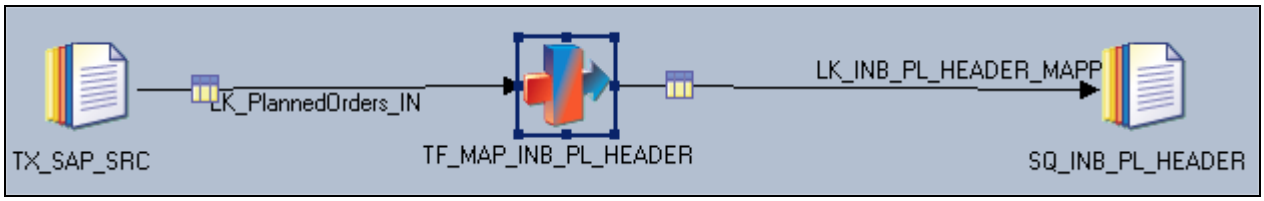


Illustration 49: Exemple de service DataStage.

7.2.1.4 Traitement ordonnancé

Le schéma ci-dessous illustre les enchainements des étapes permettant d'exécuter chaque interface d'extraction. En règle générale toutes ces interfaces sont déclenchées au même moment, en soirée lorsque l'activité sur les systèmes est réduite.

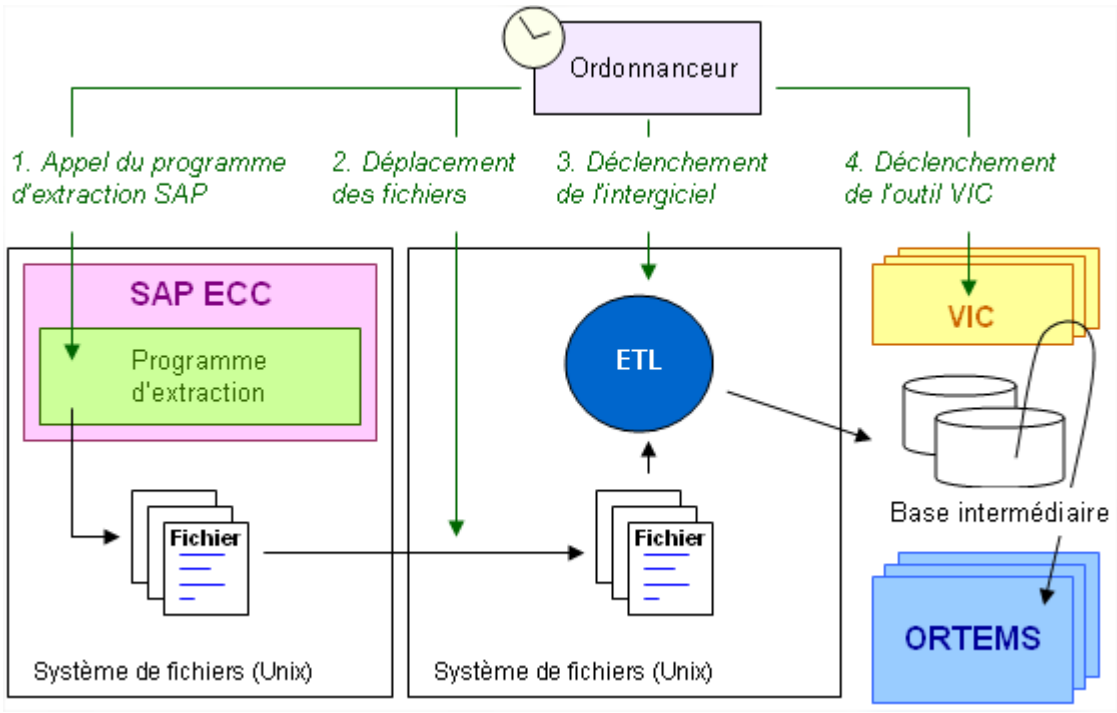


Illustration 50: Enchaînement des étapes du flux transférant les données de SAP ECC vers ORTEMS.

1. *Appel du programme d'extraction*

ControlM, l'ordonnanceur global du système informatique de Givaudan, appelle le programme d'extraction sur le progiciel SAP ECC. A la fin de son exécution, des fichiers de données sont générés sur le serveur de type Unix qui héberge SAP.

2. *Déplacement des fichiers*

L'ordonnanceur exécute des commandes système afin de déplacer les fichiers de données vers le serveur Unix sur lequel l'intergiciel DataStage est installé.

3. *Déclenchement de l'intergiciel*

Les processus DataStage sont déclenchés par ControlM. A la fin de ce traitement, les données des fichiers d'extraction sont copiées dans les tables de la base de données intermédiaire selon les règles de mapping développées.

4. *Déclenchement de l'outil VIC*

Dès lors que les traitements DataStage sont terminés, l'ordonnanceur active le processus d'intégration de VIC. Selon les règles implémentées par les équipes locales, ce dernier synchronise ORTEMS avec le contenu de la base de données intermédiaire. La fin de cette opération clot le scénario d'exportation des données de SAP vers ORTEMS.

Pour chaque site utilisant l'application ORTEMS, ces étapes sont déclenchées le soir vers 22 heures en horaire local.

7.2.2 Flux de ORTEMS vers SAP ECC

Les flux d'intégration d'ORTEMS vers SAP ECC poursuivent un cycle similaire, mais de manière inversée. Tandis que dans le scénario d'importation de données à ORTEMS l'outil VIC lisait les données dans la base de données, il doit ici lire les données dans l'application puis les insérer dans des tables dédiées de la base. Le travail de DataStage consistera ensuite à lire ces tables et écrire les données dans des fichiers texte. Ces derniers seront ensuite lus par un programme d'importation de données dans SAP qui générera automatiquement des iDocs pour traiter une à une les modifications sur les objets relatifs aux données reçues. L'usage d'iDocs en interne à SAP pour les scénarios d'intégration de type batch ajoute de la complexité aux développements mais permet de s'assurer que si le traitement d'un objet échoue, l'ensemble du processus d'intégration ne sera pas abandonné.

Les interfaces d'importation de données dans SAP ECC au travers de processus DataStage sont rares, dans la plupart des cas il est possible d'utiliser l'intergiciel WebMethods, bien plus adapté pour réaliser le travail de mapping des iDocs.

Le schéma ci-après représente les enchaînements des étapes dans le cadre d'un scénario de transfert de données depuis ORTEMS vers SAP.

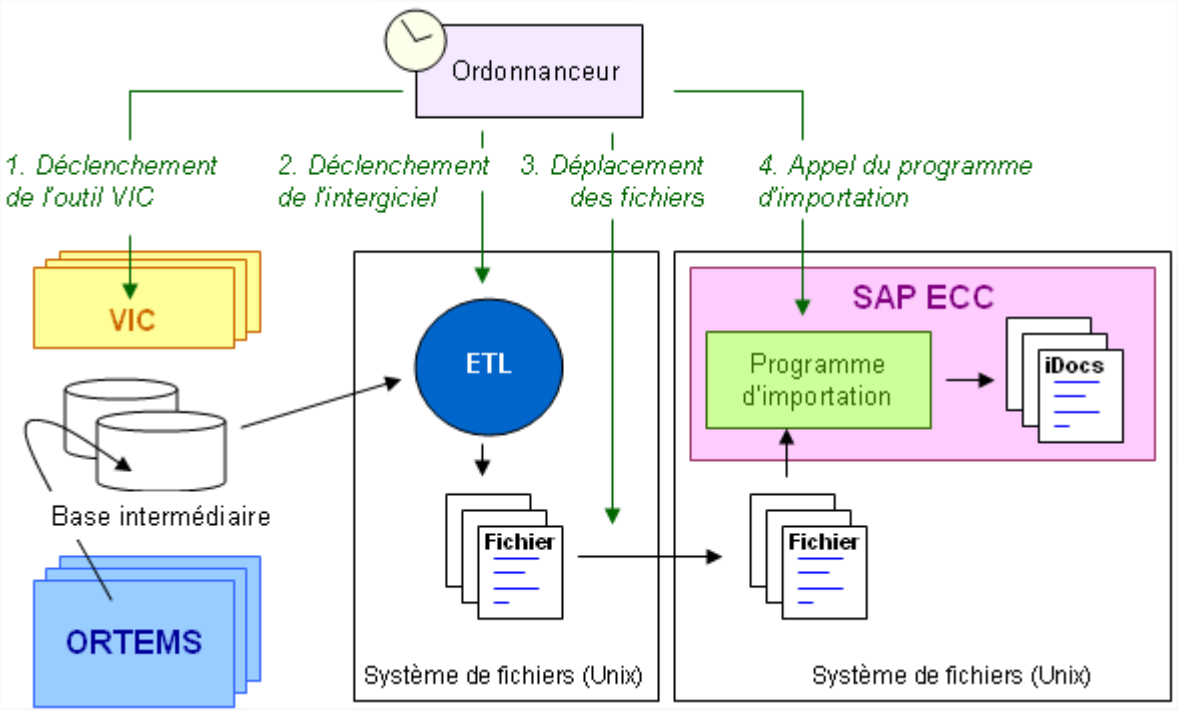


Illustration 51: Enchaînement des étapes du flux transférant des données depuis ORTEMS vers SAP ECC.

7.3 Exploitation

L'architecture choisie pour les interfaces entre SAP ECC et ORTEMS permet de répondre efficacement aux contraintes tout en disposant d'une distinction nette entre les développements locaux et globaux. Les flux conçus au cours de ce sous projet sont généralement robustes. Une fois validés dans l'environnement de développement, les erreurs sont rares. Les extractions ayant été conçues de manière la plus générale possible, c'est à dire en ne se limitant pas au transfert du strict minimum, ces interfaces n'ont presque pas eu d'extension à subir après deux ans de fonctionnement en condition de production.

8 Conception d'outils d'administration et de surveillance

Au cours de l'implémentation du progiciel de gestion intégré SAP, des besoins de développement de nouveaux automatismes et outils d'administration apparurent. Cette section présente les principales solutions qui furent conçues afin de faciliter le support fonctionnel et opérationnel.

8.1 Analyse des pointeurs de modification

Enregistrés dans la table SAP BDCP2, les pointeurs de modification sont un fantastique moyen pour analyser les raisons du déclenchement des interfaces. Fréquemment après les démarrages de nouveaux sites, des erreurs humaines liées à la méconnaissance des nouveaux processus ou à des bogues de programmation requièrent d'analyser le contenu de cette table. Néanmoins de manière standard, SAP ne dispose pas de solution permettant de faciliter cette opération: s'il n'est pas compliqué de procéder à des analyses objet par objet, il est en revanche impossible de disposer d'une vue d'ensemble sur les pointeurs créés dans le système. Pour combler ce problème, une transaction dédiée appelée ZGL_BDCP2 fut créée.

La transaction ZGL_BDCP2 dispose de trois écrans afin de naviguer dans les pointeurs de modification. Le premier, affiché dès que le programme est accédé, comporte pour chaque type de message le nombre de pointeurs existants ainsi que le total de ceux ouverts et traités. Cet écran est représenté dans l'illustration ci-après. Les lignes blanches sont celles pour lesquelles tous les pointeurs du message sont traités tandis que les lignes jaunes signifient que des pointeurs sont toujours ouverts.

Message Type	Processed	Not processed	Total
SUBMAS	78.116	18.782	96.898
WMBBIN	4	0	4
ZGLBOMMAT	624.993	6	624.999
ZGLCHPSUB	2	1.110	1.112
ZGLCMIR	20	0	20
ZGLDEBMAS	571	0	571
ZGLEHSDATA	3.738	21	3.759
ZGLHUINFO	1.215.820	402	1.216.222
ZGLMASRECIPE	79.278	45	79.323
ZGLMFGMATMAS	82.284	44	82.328
ZGLPHYSSAMPLE	10.173	10	10.183
ZGLSUBMASN	97.732	47	97.779
Z_GLCTN	371	126	497

Illustration 52: Nombre de pointeurs existants *(total)*, traités *(processed)* et modifiés *(not processed)*.

Dès lors que l'utilisateur double clique sur une ligne de ce premier tableau, une seconde liste est affichée indiquant pour chaque jour le nombre de pointeurs créés pour le message correspondant, ainsi que le total de ceux ouverts et traités. Enfin, en cliquant à nouveau deux fois sur une ligne de ce tableau, une autre liste s'affiche afin d'indiquer par utilisateur le nombre de pointeurs générés, ouverts et fermés concernant le type de message et la date correspondante.

Comme la majorité des développements SAP, cette transaction fut écrite en ABAP. Les champs de la table des pointeurs de modification BDCP2 utilisés et les requêtes SQL permettant d'obtenir les données de chacun des écrans de la transaction sont listés ci-après.

Champs	Type	Description
MESTYPE	Texte (30 caractères)	Type de message du pointeur.
PROCESS	Texte (1 caractère)	Indicateur de traitement, valeur 'X' si le pointeur est traité, vide sinon.
CREDATE	Date	Date de création du pointeur.
USRNAME	Texte (12 caractères)	Nom d'utilisateur de la personne ayant généré le pointeur au cours d'un traitement métier.

Tableau VI: Champs de la table BDCP2 utilisés pour produire les résultats de la transaction ZGL_BDCP2.

Requête 1:

Agrégation des pointeurs par état et type de message.

```
SELECT mestype, process, count(*) FROM bdcp2 GROUP BY messtype, process
```

Requête 2:

Agrégation des pointeurs par état et date de création pour un type de message donné

```
SELECT credate, process, count(*) FROM bdcp2 WHERE messtype = 'valeur sélectionnée' GROUP BY credate, process
```

Requête 3:

Agrégation des pointeurs par état et utilisateur pour un type de message et une date de création donnée.

```
SELECT usrname, process, count(*) FROM bdcp2 WHERE messtype = 'valeur sélectionnée' AND credate = 'valeur sélectionnée' GROUP BY creatdate, process
```

8.2 Mise à l'écart des iDocs invalides

Qu'ils soient en entrée ou en sortie de SAP, les iDocs en erreur dans le système ne peuvent pas toujours être retraités. En conséquence, afin de faciliter le suivi des problèmes par les équipes de support, une transaction fut conçue pour permettre de changer l'état des iDocs et ainsi de les marquer comme obsolètes. Comme représenté ci-après, la transaction ZGL_DEL_IDOC

dispose d'un écran de sélection similaire aux programmes standards de SAP qui permettent de visualiser les messages. Une fois validé, les iDocs correspondants sont affichés dans une liste, il suffit alors simplement de les sélectionner, puis de cliquer sur le bouton « Flag for Deletion » disposé en haut de l'écran.

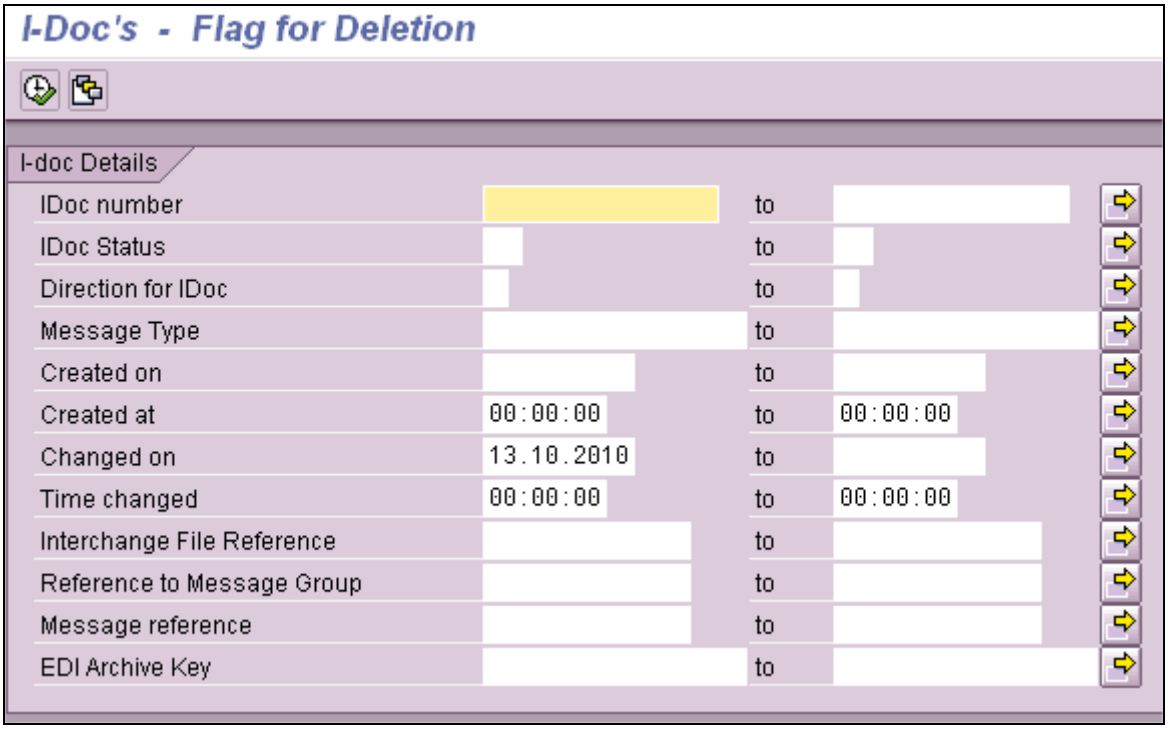


Illustration 53: Ecran de sélection de la transaction ZGL_DEL_IDOC.

Une fois marqués comme obsolètes, les iDocs en entrée et sortie du système seront respectivement disposés dans les états 68 et 31 « Erreur, arrêt du traitement ». Aucun de ces messages ne pourra alors être retraité. Néanmoins, il sera toujours possible de réaliser une copie des iDocs avec la transaction SAP WE19.

8.3 Gestion des programmes ordonnancés

La mise en place des interfaces requiert d'ordonnancer dans SAP une multitude de traitements automatisés à des fréquences d'exécution généralement très rapprochées. Les échanges d'informations ayant principalement lieu pendant les périodes d'activité sur les usines et les sites de vente, il arrive très fréquemment que des traitements soient déclenchées inutilement en dehors de ces intervalles, à l'instar de la nuit. Par défaut, le mécanisme d'ordonnancement des travaux dans SAP ne permet pas une gestion assez fine des plannings de déclenchement pour permettre de contourner convenablement ce problème. Les ressources du système se trouvent donc souvent inutilement accaparées, impactant ainsi la performance des autres processus.

Pour résoudre cet obstacle, deux programmes appelés « Cockpits » furent créés. Le premier a pour rôle d'examiner les pointeurs de modification ouverts dans le système, puis de déclencher

leur traitement en arrière plan. D'un principe similaire, le second analyse les iDocs en attente de traitement en entrée et sortie du système avant de déclencher leur exécution. Grace à eux, le nombre total de travaux ordonnancés déclenchés chaque jour fut divisé par trois sans ajouter de délais dans l'exécution des interfaces. Chacun de ces programmes devra être déclenché toutes les minutes. Ils sont décrits ci-après dans une section dédiée.

8.3.1 Cockpit de déclenchement du traitement des pointeurs de modification

Le programme de déclenchement du traitement des pointeurs de modification se configure au travers de la table dédiée ZGL_BDCP2_JOBS qui est illustrée ci-dessous. Chaque type de message pour lequel le cockpit devra vérifier l'existence de pointeurs ouverts dans le système y comporte une ligne dédiée composée des champs détaillés ci-après.

Messg.Type	Background Job Name	Lat.	Activation time	Job no.	ServerName	Active
COSMAS	FIN-INT-CP-COCKPIT_COSMAS_1H	3600	20101113151909	151909		X
CREMAS	FIN-INT-CP-COCKPIT_CREMAS_1H	3600	20101112221708	221708		X
GLMAST	FIN-INT-CP-COCKPIT_GLMAST_1H	3600	20101112125808	125808		X
LOIPRO	MFG-INT-CP-COCKPIT_LOIPRO_5M	300	20101113151909	151909		X

Illustration 54: Structure et contenu de la table ZGL_BDCP2_JOBS.

- « Message type », type de message: le nom du type de message qui doit être recherché dans la liste des pointeurs ouverts du système. Champs clef de la table.
- « Background job name », nom du traitement ordonnancé: si des pointeurs ouverts existent, alors le traitement en arrière plan devra être créé sous cette appellation.
- « Latency », latence en secondes: Durée minimum à respecter entre deux traitements. Cette valeur permet d'éviter des déclenchements trop fréquents dans les cas où les besoins métier ne justifient pas cette rapidité.
- « Activation date time », date et heure de dernière exécution: instant où le dernier programme de traitement des pointeurs de ce type de message a été déclenché pour la dernière fois.
- « Job number », numéro du dernier traitement ordonnancé: numéro système identifiant le dernier programme ordonnancé déclenché pour ce type de message. Cette information va permettre d'effectuer un contrôle afin d'éviter de déclencher une nouvelle instance du programme de traitement au cas où la dernière exécution est toujours active.
- « Active flag », état : Actif ou inactif. Si inactif, alors le type de message ne sera pas pris en considération par le cockpit de déclenchement du traitement des pointeurs de modification.

Lorsqu'il est déclenché, la première étape du cockpit est de lire les enregistrements de cette table pour lesquels il faudra vérifier l'existence de pointeurs ouverts. Il s'agit donc d'éliminer les types de messages pour lesquels la durée de latence ajoutée à la date et heure du dernier déclenchement

est inférieure à l'heure actuelle, ainsi que de ne pas sélectionner les enregistrements pour lesquels le champ état indique que la configuration est inactive.

Afin d'éviter de déclencher plusieurs fois des mêmes travaux ordonnancés alors qu'un traitement pour le même type de message est déjà en cours, le cockpit doit ensuite s'assurer que les exécutions de tous les programmes identifiés par les numéros de traitement enregistrés dans la table ZGL_BDCP2 sont terminées. Cette tâche s'accomplit au travers de modules de fonction standard de SAP.

Pour tous les types de message n'ayant pas été filtrés par les vérifications préalablement citées, la troisième étape consiste simplement à compter le nombre de pointeurs de modification ouverts enregistrés dans la table BDCP2. Chacun pour lequel ce total est supérieur à zéro doit ensuite se voir créer un travail de traitement en arrière plan dédié nommé sous l'appellation enregistrée dans la table ZGL_BDPC2_JOBS. Le cockpit met ensuite immédiatement à jour dans cette même table la date et heure du dernier déclenchement ainsi que le numéro du traitement ordonnancé.

Quelle que soit l'interface, tous les travaux en arrière plan initiés par ce cockpit déclenchent le même programme SAP avec le type de message correspondant saisi automatiquement dans l'unique zone de l'écran de sélection de ce dernier. Ainsi, il n'est pas requis de disposer dans la table ZGL_BDPC2_JOBS de champs permettant de spécifier le nom d'un programme et d'une variante dédiée au type de message.

Simple à développer, maintenir, superviser et configurer, le cockpit de déclenchement du traitement des pointeurs de modification est la bonne réponse au bon problème. Mis en place à la fin de l'année 2009, cette solution fut rapidement adoptée par les administrateurs SAP et les équipes de support car elle permet une gestion centralisée des programmes ordonnancés liés aux traitements des pointeurs.

8.3.2 Cockpit de déclenchement des programmes de traitement et de transfert des iDocs

Le cockpit de déclenchement des programmes de traitement et de transfert des iDocs est plus complexe que celui des pointeurs de modification. En effet, contrairement à ce dernier il s'agit ici de gérer les conditions d'exécution de programmes pouvant traiter plusieurs types de messages enregistrés dans des variantes au moyen de champs multi-valués. La configuration doit donc être stockée dans le système au travers de deux tables. La première enregistrera les caractéristiques des traitements en arrière plan à gérer par la solution, tandis que la seconde contiendra les conditions de déclenchement pour chacun des travaux. Les structures de ces tables sont illustrées et décrites ci-dessous.

Table ZGL_IDOC_JOBS, caractéristiques des travaux:

Background Job Name	Program Name	Variant	Lat.	Activation time	Job no.	Active	ServerName
ALL-INT-IDOC-INB_GEN	RBDMANI2	GEN_1H	3600	0	0	X	
ALL-INT-IDOC-INB_K1_(	Z_GL_BC_R_RE	GEN_5M	300	20101113153252	15325400	X	
ALL-INT-IDOC-OUT_K1_	RSEOUT00	GEN_5M	300	20101113144735	14473500	X	

Illustration 55: Structure et contenu de la table ZGL_IDOC_JOBS.

- « Background job name », nom du traitement ordonnancé: appellation qui sera utilisée pour nommer les travaux en arrière plan déclenchés en correspondance à cette ligne de configuration. Champ clef.
- « Program name » : nom du programme qui devra être exécuté par le traitement ordonnancé.
- « Variant », variante: nom de la variante (conteneur de paramètres d'un écran de sélection) qui devra être utilisée pour l'appel du programme.
- « Latency », latence en secondes: Durée minimum à respecter entre deux déclenchements. Cette valeur permet d'éviter des activations trop fréquentes dans les cas où les besoins métier ne justifient pas cette rapidité.
- « Activation time », date et heure de dernière exécution: instant où le traitement ordonnancé à été déclenché pour la dernière fois.
- « Job number », numéro du dernier traitement ordonnancé: numéro système identifiant le dernier programme ordonnancé déclenché pour cette ligne de configuration. Cette information va permettre d'effectuer un contrôle afin de déclencher une nouvelle instance du programme de traitement uniquement au cas où la dernière exécution est terminée.
- « Active flag », état : Actif ou inactif. Si inactif, alors la ligne de configuration ne sera pas prise en considération.
- Autres champs pour la gestion de l'état du traitement ordonnancé par des systèmes externes. Fonctionnalité détaillée dans le chapitre « 8.4 - Connexions SAP ECC - EAI WebMethods ».

Table ZGL_IDOC_JOBS_TR, configuration des conditions de déclenchement:

Partn.no.	Partn.no.	Message Type	Msg. ref.	Status	Background Job Name
*	K1_OTC	COND_A	*	64	OTC-INT-IDOC-INB_OTC_5M
*	K1_OTC	ZGLCMIR00	*	64	OTC-INT-IDOC-INB_OTC_5M
*	K1_OTC	ZGLCUSTHIER	*	75	OTC-INI-IDOC-INB_K1_OTC_CUSTH
*	K1_QUA	ZGLINSPOPER	*	64	QUA-INT-IDOC-INB_K1_QUA_5M
*	K1_QUA	ZGLPOOLPERI		64	QUA-INT-IDOC-INB_K1_QUA_5M

Illustration 56: Structure et contenu de la table ZGL_IDOC_JOBS_TR.

- Nom du profil de partenaire: uniquement les iDocs créés avec ce profil de partenaire seront considérés. Cette zone peut contenir « * » afin de considérer n'importe quel profil. Champ clef.
- Nom du type de message: type de message des iDocs devant être considérés par cette condition. Champ clef.
- Référence de message: champ utilisé comme filtre pour contenir le code usine ou atelier, voir chapitre « 6 - Cas 2: Gestion d'un cas de synchronisation complexe avec l'application CMS ». Peut contenir « * » pour considérer n'importe quelle valeur. Champ clef.
- Statut: numéro de l'état des iDocs devant être considérés par cette règle. Les valeurs utilisées sont principalement: 64 « en attente de traitement », 75 « iDoc en file d'attente », 30 « iDoc en attente de transfert » ou encore 51 « terminé avec erreur » afin de programmer des nouvelles tentatives d'exécution pour des messages dont le traitement a déjà échoué. Champ clef.
- Nom du traitement ordonnancé: identifiant du traitement ordonnancé enregistré dans la table ZGL_IDOC_JOBS qui devra être déclenché si cette condition est trouvée dans le système. Plusieurs conditions peuvent être affectées à un même traitement ordonnancé.

Une fois déclenché, le cockpit sélectionne tous les enregistrements de la table ZGL_IDOC_JOBS disposés en état actif et pour lesquels la durée de latence ajoutée à la date et heure du dernier déclenchement est inférieure à l'heure actuelle. Le programme vérifie ensuite que tous les traitements ordonnancés préalablement déclenchés sont terminés afin d'éviter les collisions, pertes de séquence ou double-traitements pouvant être causés par des exécutions parallèles de deux travaux identiques en arrière plan.

Pour chacun des traitements ordonnancés restants après l'application de ces filtres, le programme sélectionne ensuite les conditions de déclenchement enregistrées dans la table ZGL_IDOCS_JOBS_TR. L'étape suivante consiste à vérifier pour chacune des conditions s'il existe des iDocs correspondants dans le système. Cette opération s'exécute au moyen de mécanismes standard de SAP.

Dès lors qu'un iDoc respectant une condition est trouvé, un travail en arrière plan est déclenché selon les paramètres enregistrés dans la table ZGL_IDOC_JOBS. Comme pour le précédent cockpit, la date et l'heure de dernière exécution ainsi que le numéro du dernier traitement sont automatiquement mis à jour dans cette table. Pour éviter de consommer inutilement des ressources système, aucune autre condition affectée au même traitement ordonnancé ne sera vérifiée. L'algorithme se poursuit ensuite avec l'analyse des conditions suivantes.

Grâce à ce cockpit, le nombre de déclenchements des programmes de traitement d'iDocs fut réduit de près de 70%. En effet, avec ce mécanisme les interfaces asynchrones les plus exigeantes sur les délais de synchronisation n'ont dorénavant plus de traitements ordonnancés dédiés déclenchés toutes les minutes durant les périodes où il n'y a aucune activité.

8.4 Connexions SAP ECC - EAI WebMethods

L'intergiciel WebMethods et le progiciel SAP ECC sont interconnectés de manière permanente. Néanmoins cette liaison nécessite fréquemment d'être interrompue pour permettre l'exécution des opérations suivantes :

- Backup de l'EAI WebMethods.
- Opérations de maintenance sur l'intergiciel ou sur SAP ECC.
- Transport de nouvelles versions des programmes et interfaces, tant sur SAP ECC que sur WebMethods.

Il est important que toutes les interruptions soient gérées de manière à garantir la stabilité des interfaces. Ainsi, lorsque SAP ECC est inactif alors WebMethods doit conserver les messages dans ses files d'attente tout au long de l'indisponibilité. Dès lors que le progiciel sera réactivé, les messages en attente devront être transmis dans l'exacte séquence de leur publication, tel que si le système n'avait jamais été désactivé. A l'inverse, lorsque l'EAI est inactif alors les iDocs en sortie de SAP ECC doivent rester dans le statut 30 « iDoc en attente de transfert ». Dès que WebMethods sera de nouveau disponible, alors ces messages devront être transférés en respectant leur ordre exact de création.

La conception de mécanismes spécifiques est nécessaire pour permettre d'automatiser ces coupures et réactivations, ou encore de les rendre facilement exécutables par les administrateurs systèmes. Ces solutions sont décrites dans des paragraphes dédiés ci-après.

8.4.1 Désactivation des transferts d'iDocs en sortie de SAP ECC

Les communications entre SAP ECC et WebMethods doivent être interrompues dès lors que l'intergiciel est indisponible. Les coupures et réactivations peuvent être déclenchées par l'ordonnanceur ControlM avant et après une tâche de sauvegarde générale, ou alors directement par les administrateurs de l'EAI au cours de l'exécution d'opérations de maintenance ou de transport des nouvelles versions des développements des interfaces.

SAP ECC transfère les iDocs vers l'intergiciel WebMethods dès lors que le programme standard RSEOUT00 est exécuté. Ainsi, pour empêcher l'envoi des messages il est nécessaire d'interrompre tous les appels à ce programme. Les exécutions de RSEOUT00 étant toujours activées par le cockpit de déclenchement des traitements et transferts d'iDocs, il suffit donc d'adapter ce dernier pour y inclure de nouveaux contrôles. Les interruptions devront pouvoir être commandées par ControlM et par WebMethods tout en garantissant que l'ordonnanceur ne puisse réactiver des traitements désactivés depuis l'intergiciel, et inversement.

La mise en place de cette solution est simple. La table de configuration des traitements ordonnancés ZGL_DEL_IDOC comporte une zone « état » permettant de désactiver les exécutions automatiques. Ce mécanisme de contrôle doit être étendu par trois nouveaux champs à ajouter dans la table:

- Contrôle externe: oui ou non. Si oui, alors l'état du traitement peut être contrôlé à partir des outils externes à SAP ECC (intergiciel et ordonnanceur).
- Etat intergiciel: actif ou inactif. L'état sera positionné grâce à des outils à développer sur la plateforme d'administration de WebMethods.
- Etat ordonnanceur: actif ou inactif. Cet état est positionné par des appels à des programmes déclenchés depuis l'ordonnanceur ControlM.

Le programme du cockpit doit être modifié de manière à inclure les logiques suivantes:

- Si la zone « contrôle externe » est positionnée à « non », alors seul le champ « état » déjà présent dans la table ZGL_DEL_IDOC doit être utilisé pour déterminer si le traitement ordonnancé est actif ou inactif.
- Sinon si la zone « contrôle externe » est positionnée à « oui » (cas inverse), alors le traitement ordonnancé sera déterminé comme actif uniquement si les champs « état », « état intergiciel » et « état ordonnanceur » sont tous les trois positionnés avec la valeur « actif ».

Pour que ControlM puisse positionner adéquatement la valeur de la zone « état ordonnanceur » avant et après l'indisponibilité de l'intergiciel WebMethods, deux nouveaux programmes doivent être développés sur SAP ECC. Le premier lira la table ZGL_DEL_IDOC et positionnera à « inactif » la zone « état ordonnanceur » pour tous les traitements ayant le champ « contrôle externe » disposé à « oui », tandis que le second exécutera l'inverse en positionnant la zone « état ordonnanceur » à « actif ». Intégrés dans la chaîne d'arrêt et de redémarrage de l'intergiciel EAI, l'exécution de ces deux programmes permettra de geler et dégeler les traitements ordonnancés transférant les iDocs de SAP ECC vers WebMethods.

Les valeurs de la zone « état intergiciel » sont quant elles gérées directement à partir d'une interface web créée sur l'outil de supervision et d'administration de l'EAI WebMethods. La communication avec SAP ECC se réalise au moyen de modules de fonctions pouvant être appelés depuis l'extérieur de SAP (ces modules sont fréquemment nommés « BAPI »). La première fonction permet à WebMethods de lire les valeurs de la zone « état intergiciel » pour tous les traitements ayant le champ « contrôle externe » disposé à « oui ». La seconde et la troisième permettent quant à elles de disposer à « actif » ou à « inactif » la valeur de la zone « état intergiciel ». L'illustration ci-dessous représente l'outil conçu dans WebMethods pour permettre la gestion des états des traitements ordonnancés de SAP ECC. Les traitements sont gérés par

groupe, ici au travers du groupe « GEN » utilisé pour les interfaces couvrant les besoins de plusieurs domaines fonctionnels.

Illustration 57: Outil WebMethods de gestion des états des traitements ordonnancés de SAP ECC.

8.4.2 Désactivation des transferts de messages entre l'EAI WebMethods et le progiciel SAP ECC

La désactivation de SAP ECC nécessitant d'exécuter une multitude de tâches, une procédure spécifique a dû être conçue sur l'ordonnanceur ControlM afin d'automatiser son traitement et ainsi de limiter les risques d'erreurs. Appelée « procédure de gel », cet enchainement de tâches concerne tant le progiciel SAP ECC que les systèmes qui y sont connectés. Elle consiste à désactiver des programmes, empêcher le déclenchement de nouveaux traitements ou encore à couper des connexions. Elle est exécutée à la demande par les administrateurs SAP.

La gestion dans WebMethods de l'indisponibilité de SAP ECC a été mise en place grâce à de nouvelles étapes intégrées dans cette procédure. Des solutions pour interrompre les communications existant déjà sur l'intergiciel, il a suffit de configurer dans l'ordonnanceur ControlM un appel automatique à de nouvelles commandes lors de l'exécution des procédures de gel et de dégel. L'EAI WebMethods étant par défaut prévu pour constituer des files d'attente dès lors qu'une connexion est inactive, aucun autre développement ne fut nécessaire à la mise en place de ces règles.

8.5 Outil de gestion des transports intergiciels

La complexité des systèmes et des processus de Givaudan nécessite de développer continuellement une multitude de nouveaux programmes et interfaces, chacun de ces éléments étant régulièrement modifié dans le but de corriger les erreurs ou afin de s'adapter à l'évolution des besoins de l'organisation métier. Etant donné le volume de production de ces livrables, la gestion des transports des développements sur le paysage applicatif doit être organisée selon des règles précises supervisées grâce à des outils spécialisés. Ce rôle est la responsabilité d'une équipe dédiée appelée « release managers » (responsable des publications).

Le mécanisme de gestion des transports de SAP est simple. Chaque développement est enregistré dans un ordre de transport identifié par un numéro, puis cet ordre est ensuite transporté dans les divers environnements du paysage applicatif. Une fois exportés pour la première fois de l'environnement de développement, les ordres ne peuvent plus être modifiés. Afin de garantir le respect des règles de transport du projet et veiller à une bonne synchronisation des informations

avec l'outil Mercury Quality Center, un cockpit de gestion des ordres de transport fut créé directement dans SAP ECC.

En revanche, les intergiciels WebMethods et DataStage ne disposent par défaut d'aucun outil similaire, ni même de concepts se rapprochant de celui des ordres de transport de SAP. Ainsi, les objets impactés par chaque développement doivent être suivis attentivement afin de ne pas égarer leur transport. Un tel suivi se révélant être impossible dans le contexte exigeant du projet Outlook, un outil web appelé « TrackDev » fut conçu spécifiquement afin de faciliter la gestion des transports intergiciels, de limiter les risques d'erreurs et d'identifier les éventuelles dépendances.

Chaque développement réalisé sur les intergiciels WebMethods ou DataStage doit se voir créer une fiche sur TrackDev. Celle ci comporte obligatoirement le numéro d'identifiant Mercury correspondant au développement, la liste des objets impactés sur l'intergiciel et une description de la procédure de déploiement. Par ailleurs, grâce à une interface avec Mercury, des informations maintenues dans l'outil de gestion de la qualité sont automatiquement mises à disposition sur TrackDev. Une fois que le transport d'un développement est réalisé, alors les administrateurs des intergiciels doivent consulter la fiche correspondant au développement, puis valider la promotion du code vers l'environnement suivant sur le chemin du transport.

Un rapport spécifique fut conçu dans TrackDev à l'usage des responsables des publications. Appelé « transport buffers » ce rapport fournit une vue d'ensemble des transports en attente dans le paysage applicatif. Pour chaque environnement, la liste des développements n'étant pas présents dans l'environnement de niveau directement supérieur est affichée. Il est ainsi possible de disposer en un coup d'œil de la liste des transports restant à exécuter sur le paysage applicatif. Un exemple de fiche de transport pour un développement sur l'intergiciel WebMethods ainsi qu'une copie du rapport « transport buffers » sont consultables dans les annexes de ce document.

Conclusion

L'implémentation dans une entreprise d'un nouveau progiciel de gestion intégré provoque une révolution pour la majorité des composants de l'écosystème applicatif. La réalisation de ce projet dans la société Givaudan n'a pas dérogé à cette règle. Certaines applications, tel le système de gestion de la production CMS ont dû subir une refonte importante afin de s'adapter aux nouveaux concepts de gestion des processus métiers apportés par le progiciel SAP ECC. D'autres, à l'image de CDM, l'outil de gestion de la hiérarchie des clients, n'ont subi que de faibles ajustements pour leur permettre d'assurer la continuité du service.

Dans un tel projet, deux stratégies principales doivent alors diriger le processus d'intégration. La première, destinée aux applications ayant un avenir sur le long terme dans la société, suggère une adaptation complète des systèmes tiers au nouveau progiciel. Ainsi, ces applications doivent se voir modifiées en profondeur afin de couvrir les besoins fonctionnels de leur communication avec le progiciel, qu'il s'agisse de SAP ECC ou d'un autre système. En évitant de la sorte la construction de « ponts », la fiabilité et la pérennité de la solution développée est donc automatiquement préservée. Par conséquent, les mécanismes spécifiques à la liaison de ces systèmes avec le progiciel de gestion intégré d'origine - BPCS chez Givaudan - doivent donc être entièrement supprimés.
Plus pragmatique, mais complémentaire à la première, la seconde stratégie vise à minimiser les efforts pour l'intégration des applications ayant un futur projeté uniquement sur le court terme. En effet, la durée de vie de certains systèmes et flux de communication est parfois limitée à la phase de transition pendant laquelle tous les sites ou toutes les fonctionnalités ne sont pas encore migrés vers le nouveau progiciel. Dans te tels scénarios, des « ponts » d'intégration peuvent être conçus afin de réduire les risques et les coûts engendrés par la mise en place des interfaces.
Par ailleurs, lorsque le flou subsiste quant à l'avenir de certaines applications, les choix de stratégies doivent être déterminés en fonction d'autres critères. Les contraintes du projet et de la disponibilité des ressources ont alors une influence importante sur les décisions.

Tout au long du projet d'implémentation de SAP ECC chez Givaudan, il a fallu trouver les solutions adéquates aux contraintes de chaque cas d'intégration, valider les concepts les plus incertains au travers de prototypes et continuellement œuvrer à réaliser des flux réutilisables afin de ne pas redévelopper la roue lors de la conception de chaque nouvelle interface. Plus de deux ans après le premier démarrage, les standards d'intégration déterminés au cours de la première phase d'analyse du projet n'ont que peu eu à évoluer, malgré les conditions d'usage extrêmes de quelques types d'échanges. Néanmoins, la conception tant technique que fonctionnelle de certaines interfaces a parfois du être inévitablement revue. En effet, l'évolution progressive de notre compréhension du système, les changements des besoins de l'organisation métier et les exigences importantes de l'environnement de travail du projet Outlook ont parfois amené à devoir réajuster les objectifs.

A travers plus de soixante interfaces conçues pour échanger des données en entrée et en sortie du progiciel SAP ECC, mon travail sur l'intégration m'a permis d'acquérir et de confirmer des connaissances techniques très diverses. Les mécanismes ALE, la manipulation des iDocs dans SAP et la subtilité de certains processus d'intégration standard du progiciel de gestion intégré ont progressivement perdu leurs secrets. Les possibilités offertes par les intergiciels WebMethods et DataStage ainsi que la flexibilité apportée par ces derniers dans les processus opérationnels et fonctionnels, ont fait de moi un fervent soutient à l'usage et au déploiement des solutions standardisées. J'ai de plus découvert une multitude de système tiers de Givaudan que pour lors je ne connaissais que de nom. Par ailleurs, des contraintes de ressources m'ont parfois amené à devoir veiller moi même à la réalisation de la partie applicative non-SAP des interfaces, quelles que soient les technologies sur lesquelles ces systèmes étaient fondés. Enfin, dans un projet constitué d'une équipe formée pour la plupart de consultants ne connaissant pas l'organisation Givaudan, mon rôle interne à l'équipe d'implémentation mais en relation continue avec les divers groupes du service informatique de la société, a permis de garantir la conformité des solutions et permettre un dialogue facilité entre les acteurs concernés.

Outre l'acquisition de multiples nouvelles connaissances techniques, cette expérience riche m'a permis de découvrir l'écosystème des grands projets informatiques. Par ailleurs la confiance totale que m'a accordée mon supérieur - le responsable de l'équipe technique - fut pour moi l'occasion de montrer ma capacité à prendre des responsabilités. Ce rôle impliquant une communication constante avec de nombreux interlocuteurs, il m'a été nécessaire d'apprendre à convaincre, organiser et gérer des réunions, former des personnes, déléguer des tâches ou encore travailler à distance avec des centres de développement.

Glossaire

ABAP: Advanced Business Application Programming. Langage de développement informatique dérivé du COBOL utilisé sur les produits SAP.
ALE: Application Link Enabling. Ensemble d'outils permettant de faire communiquer les progiciels SAP avec des systèmes externes.
B2B: Business to Business. Echanges électroniques entre les entreprises en vue de réaliser des transactions commerciales.
BAPI: Business Application Programming Interface. Fonctions SAP pouvant être appelées depuis des systèmes externes.
BDM: Business Data Model. Structure de données métier utilisée pour échanger des informations entre des applications au travers d'un intergiciel.
BPCS: Business Planning and Control Système. Progiciel de gestion intégré édité par la société Infor Global Solutions.
CDM: Customer Data Management. Application développée par Givaudan qui permet de gérer les processus de fabrication dans les usines.
CMS: Compounding Management System. Application développée par Givaudan qui permet d'administrer les données de la hiérarchie des clients.
ControlM: Outil édité par la société BMC Software permettant d'ordonnancer l'exécution de tâches sur différents types de systèmes.
DataStage: Intergiciel de transfert de données par lots (batch). Edité par IBM.
EAI: Enterprise Application Integration. Famille d'intergiciels d'échange rapides et de manière asynchrones de messages électronique de faible taille.
ETL: Extract Transform & Load. Famille d'intergiciels d'échange de données par lots. Les ETL sont utilisés pour les échanges massifs n'ayant pas une contrainte de latence stricte.
IDoc: Intermediate Document. Message asynchrone reçu ou envoyé depuis un système SAP.
Intergiciel (middleware): Famille d'outils d'intégration permettant de gérer la communication entre les systèmes.
Mercury Quality Center: Outil de gestion de la qualité édité par la société HP.
Module de fonction: Fonction SAP développée dans le langage ABAP.
Mapping: Mise en correspondance de valeurs issues de structures de données différentes.
ORTEMS: Application permettant de réaliser des plans de planification pour l'exécution de processus de fabrication industriels. Editée par la société ORTEMS.
Pointeurs de modification: Mécanisme SAP permettant de tracer les créations, modifications et effacements d'objets métiers.
PGI: Accronyme de «Progiciel de gestion intégré »
Progiciel de gestion intégré: Logiciel de gestion des processus opérationnels d'une entreprise.
RFC: Remote Function Call. Protocole SAP de communication synchrone avec des applications externes.
SAP AG: Systems, Applications and Products in data processing. Editeur de progiciels.

SAP ECC: Progiciel de gestion intégré édité par SAP AG. ECC signifie « Enterprise Central Component ». Par abus de langage, SAP ECC est communément appelé SAP.
SAP GUI: SAP Graphical User Interface. Logiciel à installer sur les postes de travail pour permettre l'accès aux systèmes SAP.
Segment: Sous structure d'un type iDoc. Les segments contiennent une suite de zones dont la taille maximale ne peut dépasser mille caractères.
SGBD: Système de Gestion de Bases de Données. Ensemble de logiciels permettant de manipuler des bases de données.
SOAP: Simple Object Access Protocol. Protocole de communication d'objets entre systèmes distants.
SQL: Structured Query Langage. Langage informatique normalisé permettant de réaliser des opérations sur des bases de données,
Stream: compléter
Type d'iDoc (iDoc type): Formats de données des iDocs SAP utilisés pour les échanges électroniques asynchrones entre les systèmes. Dans le système, un type d'iDoc est une référence à une hiérarchie de segments.
Type de message (message type): Référence descriptive du contenu d'un iDoc.
User-exit: Emplacements laissés libres dans les programmes et fonctions standard de SAP afin de permettre la personnalisation du système au travers d'ajouts de code en langage ABAP.
VIC: Visual Interface Configurator. Outil composant de l'application ORTEMS qui permet de réaliser des intégrations avec des systèmes externes.
WebMethods: Intergiciel de type EAI édité par la société allemande Software AG.
WSDL: Web Service Description Language. Format de description normalisé des mécanismes d'accès et structures de données en entrée et sortie d'un service web.

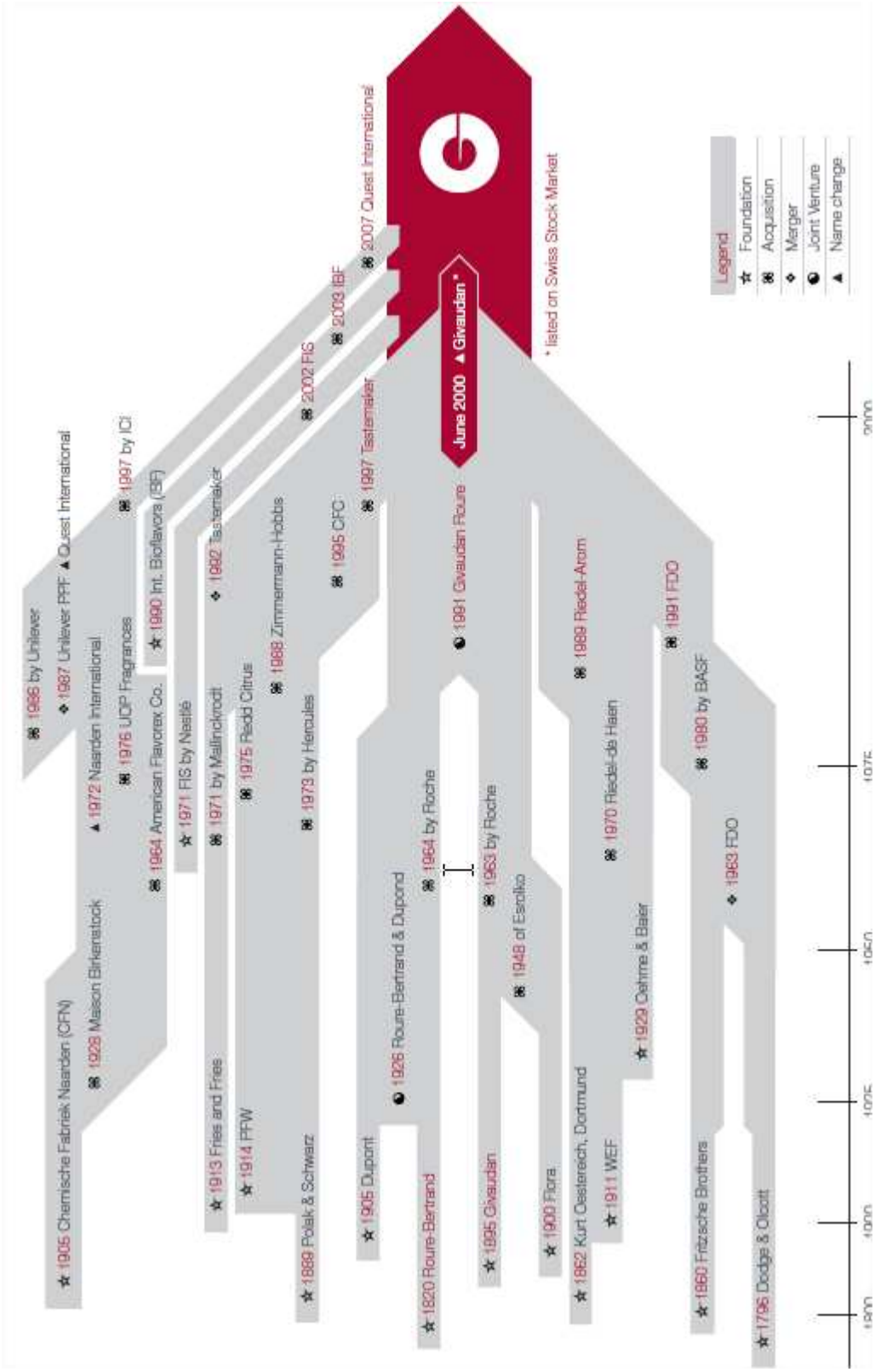
Transactions SAP

Les principales transactions SAP abordées au cours de ce document sont listées ci-dessous.

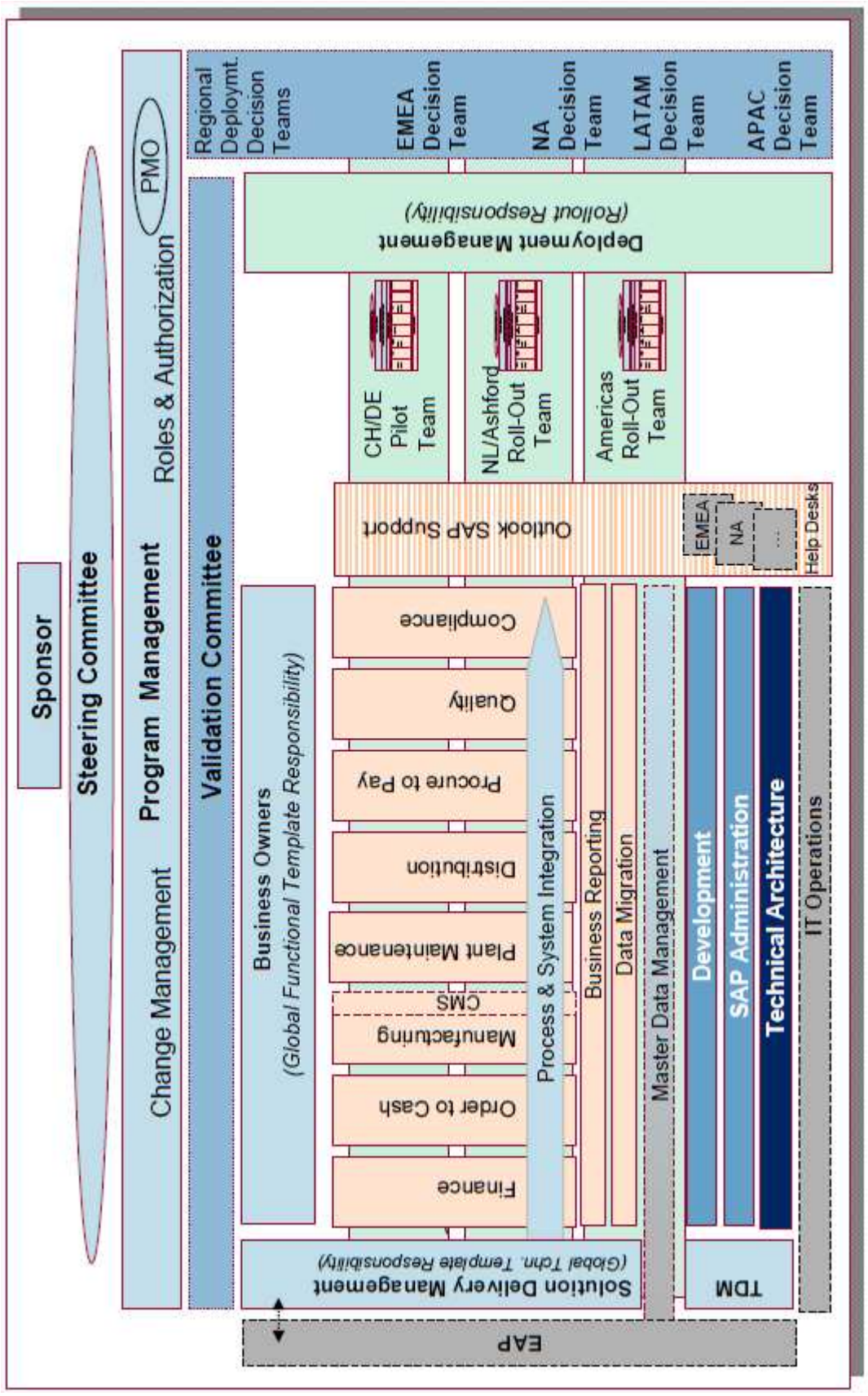
VD01: Création d'un nouveau client dans le système.
VD02: Affichage des données d'un client avec possibilité de mise à jour des informations.
VD03: Affichage des données d'un client.
VDH2N: Affichage de la hiérarchie des clients.
WE05: Liste des messages iDocs reçus et envoyés depuis un système SAP.
WE10: Recherche de messages iDocs en fonction de leur contenu.
WE20: Configuration des profils de partenaire.
WE30: Créer et édité un type d'iDoc ou une extension d'iDoc.
WE31: Créer ou édité le format d'un segment.
WE42: Création d'un code de processus.
WE81: Création d'un type de message.
WE82: Affection d'un type de message à une extension et un type d'iDoc.
WEINBQUEUE: Visualisation et administration des messages iDocs reçus dans des files d'attente.
BD87: Affichage des messages iDocs reçus et envoyés depuis SAP. Agrégation par état et possibilité de retraiter les transactions.
SE11: Dictionnaire des structures et types de données SAP.
SE16N: Affichage des données contenues dans les tables SAP.
SE38: Exécution de programmes SAP.
SE80: Environnement SAP de développement en ABAP.
BD21: Déclenchement du traitement des pointeurs de modification.
BD53: Détermination des tables et champs devant déclencher l'écriture de pointeurs de modification pour chaque type de message.
BD64: Configuration du modèle de distribution.
BD60: Définition des modules de fonction à appeler par SAP pour traiter les pointeurs de modification d'un type de message.
WSADMIN: Administration des services web.
WSCONFIG: Configuration des services web.

Annexes

Annexe A: Histoire de Givaudan au travers des fusions-acquisitions



Annexe B: Structure organisationnelle du projet Outlook



Annexe C: Fiche TrackDev pour le transport des développements intergiciels

Development id: **#367**, Type: **Defect**, Project: **Outlook**

Tracking id (Mercury): **33726** Show Mercury entry
RICEF **otc017**, planned for release **EF**, status **50 Defect closed** assigned to **pattabij**.

Description: **InboundSalesOrder, change ids for BPCS messages**

Deployment status			
Environment	Status	Executed on	Executed by
Development	Loaded	2010-02-08 10:45:06	robialb
Quality	Loaded	2010-02-08 14:19:43	mayjonal
Business Simulation	Loaded	2010-02-09 16:35:57	mayjonal
Production	Loaded	2010-02-19 07:53:44	mayjonal

The landscape promotion of this development is finished.
Show deployment grid for this development and its related components
Validate promotion to a non-landscape environment.

Linked components

SAP ECC wM Subscribe *InboundSalesOrder* Show deployment instructions for this component

Landscape dependencies:
- Business Simulation - New development #460 - Outlook (Mercury 35475)
- Production - New development #51 - Outlook (Mercury 7454)
- Production - Defect #111 - Outlook (Mercury 12506)
- Production - Change request #114 - Outlook (Mercury 12538)
- Production - Defect #121 - Outlook (Mercury 15961)
- Production - New development #240 - Outlook (Mercury 28431)
- Production - Change request #356 - Outlook (Mercury 33259)

SAP ECC wM Subscribe *SalesOrder* Show deployment instructions for this component

Landscape dependencies:
- Production - New development #45 - Outlook (Mercury 2928)
- Production - Defect #116 - Outlook (Mercury 13200)

Deployment procedure:

Package CK_ECC_GEN
CK_ECC_GEN.InboundSalesOrderSubscribe:ZORDERS05BSend
CK_ECC_GEN.SalesOrderSubscribe:ORDERS05Send

Can be promoted anytime

Annexe D: Rapport TrackDev « transport buffers », liste des transports en attente

Buffer of environment Development

Development	Short text	Mercury	RICEF	Release	Status
New development #**347**	SAP CUA subscribe to UserMasterData, mapping change	**33080**		4.71	05 Ready for Test in Dev
Defect #**567**	Add AccountGroup Z003 to filterfor custMaint	**48366**	OTC056	4.72	09 Successfully Tested in DEV
New development #**569**	add email addresses for payment rejects from citiBank	**48679**		4.72	09 Successfully Tested in DEV
New development #**548**	Send ZORDERS05B idoc to Exel	**47233**		4.73	05 Ready for Test in Dev
New development #**434**	FragranceFormula publish - Delete notifications where FMSEQ is null	**37932**	MFG011	New	50 Defect closed

Buffer of environment Quality

Development	Short text	Mercury	RICEF	Release	Status
Defect #**511**	NOAM_Check Outsourcing	**44857**		TBD	25 Ready for Test in QAS

Buffer of environment Business Simulation

Development	Short text	Mercury	RICEF	Release	Status
New development #**533**	OB10, Unilever requested to have all information on the invoice.	**45746**		4.72	35 Ready for Test in BUS
Change request #**553**	Entries created in COFRPRD table for non FL materials	**47698**	COM4031	4.72	39 Successfully Tested in BUS
Defect #**255**	US10 and CA10 filter for ECC/HR FIN interfaces	**30059**		TBD	35 Ready for Test in BUS
New development #**506**	setup Cession price for US FL	**33654**	FIN016	TBD	35 Ready for Test in BUS
Change request #**467**	US FL Sales report for Final Customers (xOTL)	**34317**	FIN002	TBD	35 Ready for Test in BUS
New development #**460**	EDI X12 Inbound Sales Orders	**35475**	OTC4086E	TBD	05 Ready for Test in Dev
Defect #**435**	Webmethods PRD, package CK_CPT	**37969**	FIN016	TBD	05 Ready for Test in Dev
New development #**450**	Setup CMS for US FL (plants EH01, CA01, DV01 and ED01).	**38643**		TBD	45 Ready for Test in PRD
Defect #**564**	Error in webMethods when non-english Risk phase long text is present in Product Safety data	**48417**		TBD	09 Successfully Tested in DEV

Bibliographie

Guides et publications

« SAP et ABAP », *Yann Szwec, A. J. Wilson,* Aout 2007.

« ALE, EDI & IDoc, 2nd Edition », *Arving Nagpal, John Pitlak,* Mai 2002.

Sites internet

Encyclopédie Wikipedia	http://fr.wikipedia.org
Givaudan	http://www.givaudan.com
SAP community network	http://www.sdn.sap.com
Software Top 100	http://www.softwaretop100.org/

Liste des illustrations et tableaux

Illustrations

Tableaux