

Synchroniser eZpublish et TinyERP avec XML-RPC

A l'heure où le e-commerce est en plein essor, les sites marchands doivent être à la pointe de l'innovation pour conquérir les marchés et conserver leurs marges. La rencontre entre CMS et solution ERP matures constitue indéniablement l'une des clés majeures pour permettre aux cybermarchands de développer de nouveaux avantages concurrentiels.

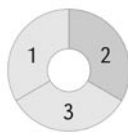
Cet article explique :

- Nous allons voir comment faire communiquer les applications Open Sources eZpublish, CMS réalisé en PHP/MySQL, et TinyERP, ERP réalisé en Python/PostgreSQL.

Ce qu'il faut savoir :

- Il vous faudra bien connaître le langage PHP, les méthodes objets, le fonctionnement des applications eZpublish et Tiny ERP, ainsi que le protocole HTTP.

Niveau de difficulté



Synchroniser eZpublish et Tiny

eZPublish est un gestionnaire de contenu *Open Source*, développé en PHP, s'appuyant sur une base de données MySQL. eZ publish est une solution totalement modulaire et personnalisable à volonté permettant la création d'un site internet évolutif, adaptée à la publication sur Internet, au e-commerce ou encore à la mise en place d'intranet. eZ publish est en fait un véritable Framework applicatif qui permet la réalisation de projets complexes. Sa structuration et son puissant modèle de contenu orienté objet, permettent de gérer les problématiques de gestion de contenu les plus ardues. Il est utilisé dans notre exemple pour créer un site internet de vente en ligne, présenter des produits, leurs prix, et permettre aux internautes d'ouvrir un compte client et de commander.

TinyERP est développé en python et s'appuie sur une base de données PostgreSQL. C'est un logiciel *Open Source* de gestion commerciale et comptable très complet, qui dispose de nombreux modules pour aider les entreprises à mieux gérer leurs ressources, qu'elles soient

financières, matérielles ou humaines. Il est notamment utilisé pour la gestion des stocks produits, le suivi en temps réel du circuit logistique et/ou des chaînes de production en son intégralité, les relations clients par le biais d'un module CRM avancé, ainsi que la gestion commerciale et financière jusqu'à l'édition d'un plan comp-

table général. Il s'agit d'une solution hautement personnalisable capable de s'adapter à tout type d'entreprise de petite et moyenne taille.

Si l'on veut utiliser les fonctionnalités de ces deux outils pour manipuler des données communes, il se pose alors un problème de synchronisation. Comment travailler en temps réel avec deux bases de données différentes, sur des informations qui doivent être à jour simultanément dans les deux applications ?

La solution utilisée dans notre cas s'appelle XMLRPC (*eXtensible Markup Language Remote Procedure Call*), un protocole de discussion permettant des appels de fonction à distance, et utilisant la norme XML pour échanger les données.

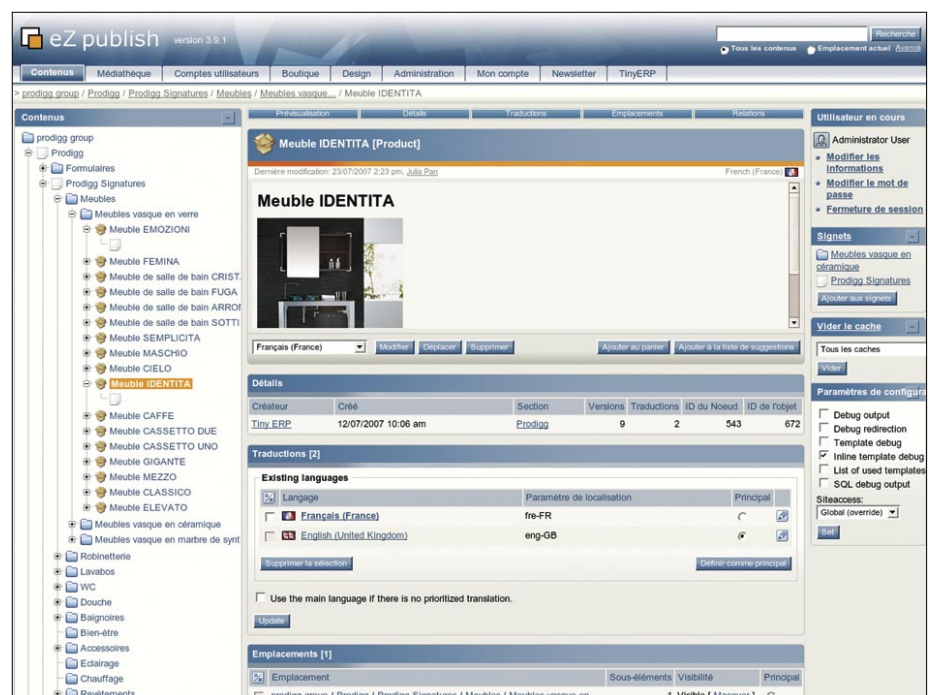


Figure 1. Interface d'administration d'eZpublish 3.9, avec l'extension Tiny ERP

Les fonctionnalités du connecteur eZ publish 3.6 – Tiny ERP 3.3 (v1)

La première version d'une solution interconnectant la version 3.6 d'eZ publish (compatible également sur les versions supérieures jusqu'à la 3.8) et la version 3.3 de Tiny ERP, a été mise en place en juillet 2006 par Internethic pour la société Pokershop. Le connecteur ainsi développé fonctionne principalement de eZ publish vers Tiny ERP, et permet notamment :

- La création et la mise à jour automatique des produits et quantités disponibles, saisis dans eZ publish et importés dans Tiny ERP.
- L'import et la mise à jour des bases de données clients d'eZ publish vers Tiny ERP : création automatique des comptes clients, rattachées ou non à une société d'appartenance, des adresses associées (facturation, livraisons), de leurs comptes comptables respectifs avec rattachement à chaque compte client de ses commandes et factures.
- L'import des commandes multi statuts et multi paiement : payée par carte bancaire ou par chèque, en attente de confirmation de livraison ou d'approvisionnement le cas échéant, avec un *workflow* de changement d'état de la commande de *en attente à en cours de traitement* dans eZ publish, associé à une notification automatique au client final par courriel.
- La création automatique des factures au sein de Tiny ERP, avec les frais de ports afférents, calculés au coût le plus avantageux, et tenant compte des remises attribuées aux clients disposant de codes promotion au sein du module de gestion de remises d'eZ publish.
- La validation des ordres d'approvisionnement entraînant la validation automatique de la demande d'expédition des colis.
- La génération le cas échéant des bordereaux d'expédition, grâce à une interconnexion avec les logiciels Expéditeur Inet ou Chronopost.
- Le réimport des numéros de colis créés, associés avec le numéro de facture généré par Tiny ERP.
- La mise à jour des stocks disponibles.
- La validation définitive de la commande avec changement d'état (*Livré*) dans le process d'achat d'eZ publish, et le rapatriement du numéro de facture et de colis au sein du compte du client.

Cette première version du connecteur est librement téléchargeable sur le site officiel d'eZ systems, dans la rubrique Communauté/Contributions.

La connexion entre eZpublish et TinyERP

Nous allons étudier dans cet article comment synchroniser les comptes clients depuis eZ pu-

blish 3.9 vers Tiny ERP 4.1 (Listing 1). Le code présenté ici n'est qu'une petite partie d'une extension bien plus importante venant se placer au sein du répertoire *extensions d'eZ publish*, et intervenant lors d'un évènement de *workflow* de celle-ci.

Les internautes vont naviguer sur le site internet conçu avec eZpublish, et s'y inscrire pour passer leurs commandes. Nous voulons alors récupérer ces informations et les ajouter dans TinyERP, en y créant leur compte de manière automatique.

Nous créons pour cela, dans eZpublish, une classe nommée TinyERP, que nous allons utili-

ser pour communiquer avec TinyERP. La première méthode de cette classe est la méthode *sendToTiny*, qui va nous servir principalement à surveiller les demandes de connexion à TinyERP, pour archiver dans une table spécifique les problèmes de connexion, le cas échéant.

Le client XML-RPC

Pour établir la connexion au serveur XML-RPC de TinyERP, nous utilisons la fonction *xu_rpc_http_concise*, proposée dans la librairie *xmlrpc_emu.inc*, disponible ici : <http://xmlrpc-epi.sourceforge.net/>

Listing 2. La fonction *xu_rpc_http_concise*

```
/* fonction xu_rpc_http_concise
 * $params contient les parametres dans un tableau de type 'variable => valeur'
 */
// On recupere les variables $method, $host, $uri, $port, et $args
extract($params);
// On prepare notre requete au format xml
$request_xml = xmlrpc_encode_request( $method, $args, array( version => 'xmlrpc' ) );
// On envoie les donnees au serveur XMLRPC avec la methode POST du protocole HTTP
$response_buf = xu_query_http_post( $request_xml, $host, $uri, $port );
// On recupere le resultat a l'intérieur de la reponse au format xml
$return = find_and_decode_xml($response_buf, $debug);
// On retourne le resultat
return $return;
```

Listing 1. La fonction *sendToTiny*

```
/* fonction sendToTiny
 * $config contient les informations de connexion a TinyERP
 * $class contient la classe d'objet de TinyERP que nous souhaitons manipuler
 * $function contient la methode appelee dans cette classe
 * $values contient un tableau de valeurs avec les parametres.
 */
// On prepare les donnees
$args = array( $config['bdd'], // la base de donnees utilisee
              $config['uid'], // l'identifiant de connexion
              $config['pwd'], // le mot de passe de connexion
              $class, // la classe d'objet
              $function, // la methode appelee
              $values) // les parametres

// On prepare la requete
$request = array( 'method'=> "execute", // la methode XML-RPC appelee
                'host' => $config['host'], // le serveur TinyERP
                'uri' => '/xmlrpc/object', // l'url du serveur XML-RPC de TinyERP
                'port' => $config['port'], // le port de connexion au serveur TinyERP
                'args' => $args) // les donnees preparees

// On envoie notre commande
$result = xu_rpc_http_concise( $request );
// Si l'envoi echoue
if ( is_array( $result ) && isset( $result['faultCode'] ) ) {
    // On enregistre l'erreur
    TinyERP::addToLog( 'error', 'Erreur : '.$result['faultString'] );
    TinyERP::addToLog( 'error', 'Classe : '.$args[3] );
    TinyERP::addToLog( 'error', 'Fonction : '.$args[4] );
    TinyERP::addToLog( 'error', 'Valeur : '.print_r( $args[5] ) );
    $result = null;
}
// On retourne le resultat
return $result;
```

Dans cette fonction, nous préparons un bloc de données XML à l'aide de la fonction PHP `xmlrpc_encode_request`. Nous envoyons ensuite ce bloc au serveur distant en utilisant la fonction `xu_query_http_post`. Nous récupérons alors la réponse dans le bloc XML retourné par le serveur à l'aide de la fonction `xmlrpc_decode` (Listing 2 et 3).

La fonction `xu_rpc_http_concise` appelle la fonction `xu_query_http_post` pour établir

la connexion au serveur http, transmettre la requête xml, et récupérer le résultat en xml. Nous disposons maintenant d'une méthode pour communiquer avec TinyERP.

La synchronisation de la création d'un utilisateur

Voyez maintenant comment utiliser cette méthode pour créer un nouvel utilisateur sous Tiny (Listing 4). Lors de l'inscription d'un uti-

lisateur sous eZpublish, le moteur d'eZpublish va publier un nouvel objet de type utilisateur. Vous allez intercepter cet appel à publication à l'aide d'un déclencheur, prévu dans eZpublish, du type `content_publish_after`, qui signifie, littéralement, après la publication d'un objet. Vous êtes donc en mesure de déclencher une action après la publication d'un objet. Vous devez alors créer un *workflow*, sous eZpublish, qui sera associé à ce déclencheur. Notre *workflow* aura pour rôle de filtrer le type d'objet publié, pour ne s'exécuter que sur la publication d'un utilisateur, et de faire appel à la fonction `createUser`.

La création du compte comptable associé

La méthode `createAccount` va nous permettre d'associer chaque partenaire à un compte comptable (Listing 5). Dans un plan comptable général, les créances liées à la vente de biens ou services rattachés au cycle d'exploitation sont enregistrées au compte 41 : *Clients et comptes rattachés*, et, plus précisément, les comptes *Clients* au compte 411.

Nous allons classer nos comptes clients par ordre alphabétique, selon la première lettre du nom du partenaire. Nous obtenons ainsi tous les noms commençant par *A*, puis tous les noms commençant par *B*, etc... Pour conserver cette organisation, nous enregistrons tous les partenaires en *A* au compte 41101, tous les partenaires en *B* au 41102, etc. jusqu'aux partenaires en *Z*, enregistrés au compte 41126. Les numéros de comptes sont ensuite incrémentés de manière automatique en partant de 000000 pour chacune des lettres, ce qui laisse une possibilité d'un million de client par lettre, soit près de 26 millions au total.

Périmètre d'application du code présenté

Lors de l'inscription d'un client sur le site eZpublish, nous avons maintenant un processus de création automatisée du même utilisateur dans TinyERP, conservant son nom, son prénom, son adresse mail, son adresse postal. La relation entre les deux utilisateurs existe dans les deux applications, le nom du partenaire TinyERP contenant l'id de l'utilisateur eZpublish, et l'utilisateur eZpublish contenant un attribut `erp_id`. Nous avons ainsi répercuté la création d'un utilisateur eZpublish dans TinyERP, et automatisé la création de son compte comptable associé. La méthode utilisée nous permettra par la suite de synchroniser également les commandes, ainsi que les demandes de renseignements. Toute la gestion du client pourra ainsi être faite au sein de TinyERP.

Mais nous avons décrit uniquement le processus de synchronisation des comptes clients d'eZpublish vers Tiny ERP. La mise en place du connecteur en mode inversé, synchronisant la création des comptes utilisateurs depuis Tiny

Listing 3. La fonction `xu_query_http_post`

```

/* fonction xu_query_http_post
 * On retrouve les variables $request_xml, $host, $uri, $port
 */
$response_buf = "";
if ($host && $uri && $port) {
    // On compte le nombre de caracteres contenus dans notre bloc de donnees xml
    $content_len = strlen($request_xml);
    // On ouvre une connexion au serveur $host sur le port $port
    $query_fd = $fsockopen($host, $port, $errno, $errstr, 10);
    // Si la connexion est ouverte
    if ($query_fd) {
        // On prepare la requete que l'on va envoyer au serveur http
        $http_request =
            "POST $uri HTTP/1.0\r\n" .
            "User-Agent: xmlrpc-epi-php/0.2 (PHP)\r\n" .
            "Host: $host:$port\r\n" .
            "Content-Type: text/xml\r\n" .
            "Content-Length: $content_len\r\n" .
            "\r\n" .
            $request_xml;
        // On envoie notre requete au serveur
        fputs($query_fd, $http_request, strlen($http_request));
        // On recupere la reponse du serveur ligne par ligne
        $header_parsed = false;
        $line = fgets($query_fd, 4096);
        // Tant que l'on recupere un reponse du serveur, on lui redemande une ligne
        while ($line) {
            // Le debut de la reponse ne nous interesse pas,
            // ce sont les entetes du serveur.
            // On ajoute la ligne a notre tampon de reponse
            // uniquement apres avoir rencontre une premiere ligne vide.
            if (!$header_parsed) {
                if ($line === "\r\n" || $line === "\n") {
                    $header_parsed = 1;
                }
            } else {
                $response_buf .= $line;
            }
            // On demande la ligne suivante
            $line = fgets($query_fd, 4096);
        }
        // On ferme la connexion au serveur
        fclose($query_fd);
        // Si la connexion echoue
    } else {
        $response_buf = false;
    }
}
// On retourne le bloc xml contenu dans la reponse du serveur
return $response_buf;

```

ERP vers eZ publish, est quant à elle nettement plus complexe.

Améliorations et nouveautés du connecteur eZpublish 3.9 – Tiny ERP 4.1 (v2)

Depuis avril 2007, Prodigg Group, une société spécialiste de la distribution en bâtiment, voulait développer une solution informatique plug-and-play de référence en VAD/VPC, résolument tournée vers l'e-commerce haut de gamme. Prodigg avait besoin d'optimiser ses process de vente et dans le cadre du lancement de son activité de vente à distance. Elle était à la recherche d'une solution informatique intégrée et puissante capable de gérer de façon interconnectée tous ses processus de vente directe en multicanal, aussi bien pour ses marchés B2B que B2C : gestion des fichiers et de la relation client, actions de prospection et de fidélisation, organisation optimisée de la logistique, automatisation et suivi des achats, assistance au process de vente du devis à la facture, analyse stratégique des données de l'activité, gestion comptable et financière.

La problématique posée par Prodigg, comparativement à celle de Pkershop, est donc bien plus complexe car elle résulte d'une organisation internationale globale répartie sur plusieurs sites : bureau de sourcing, cellule logistique, équipe commerciale, direction financière. Chaque département a un rôle et des responsabilités précises dans le déroulement, étape par étape, des process d'approvisionnement et de distribution. Les droits d'accès sur les différentes bases de données doivent donc être gérés très finement.

Un des objectifs majeurs est de pouvoir adopter une politique de gestion de stock en flux tendus, tout en continuant à offrir à ses clients une disponibilité optimale sur les produits de son catalogue. Le souhait derrière cette approche est d'offrir en temps réel au client le meilleur couple prix/délai possible en fonction de la position de chaque produit au sein de la chaîne logistique. Pour cela, l'idéal est de mettre en vente les produits, non pas lorsqu'ils rentrent en stock, mais dès que l'ordre de fabrication est lancé auprès du fournisseur. Le distributeur dispose alors de toute la durée de fabrication et de transport de la marchandise pour la vendre, à condition de pouvoir s'engager auprès de ses clients sur des délais calculés en temps réel et de leur faire profiter de conditions tarifaires d'autant plus avantageuses que la commande est passée tôt dans la chaîne logistique. Un concept de vente innovant : plus vous commandez tôt, plus vous économisez !

Il s'agit donc de mettre en place un système d'information permettant de suivre au fil de la chaîne logistique, la disponibilité et le délai de livraison d'un produit appartenant à un lot de fabrication donné, puis embarqué à bord d'une unité logistique.

Les fonctionnalités du connecteur eZpublish – Tiny ERP (v2)

Globalement, les fonctionnalités d'import et de mises à jour des données client (état civil, adresse de facturation, livraison, statut,...), implémentées dans le connecteur précédent, ont été considérablement améliorées

: désormais la connexion s'opère dans les 2 sens entre eZ publish et Tiny ERP. Il en est de même pour tout ce qui a trait au traitement automatique des commandes, génération des factures, des bordereaux d'expédition, et à la remontée des informations afférentes au sein des espaces clients d'eZ publish. Des améliorations

Listing 4. la fonction createUser

```

/* fonction createUser
 * $userId contient le numero de l'utilisateur sous eZpublish
 */
$result = $addresses = false;
// On recupere l'objet contenant notre utilisateur eZpublish
$contentObject =& eZContentObject::fetch( $userId );
// On recupere les attributs de cet objet
$contentObjectDataMap = $contentObject->dataMap();
// l'attribut user_account est lui meme un objet de type eZUser
$user_account = $contentObjectDataMap['user_account']->content();
// On recupere ainsi l'adresse mail de notre utilisateur
$userEmail = $user_account->Email;
// On recupere le pays de notre utilisateur eZpublish
$country = $contentObjectDataMap['pays']->content();
// On l'ecrit avec la premiere lettre en majuscule et le reste en minuscule
$country = ucwords( strtolower( $country ) );
// On va chercher l'id du pays correspondant dans TinyERP
$country_id = TinyERP::getCountry( $country );
// On construit le nom du partenaire que l'on va creer dans TinyERP
$last_name = strtoupper( $contentObjectDataMap['last_name']->content() );
$first_name = ucwords( strtolower( $contentObjectDataMap['first_name']->content() ) );
$name = $last_name.' '.$first_name; // $name est de la forme 'NOM Prenom'
// On prepare l'adresse de l'utilisateur
$address = array( 'name' => $name,
    'street' => $contentObjectDataMap['street']->content(),
    'zip' => $contentObjectDataMap['code_postal']->content(),
    'city' => $contentObjectDataMap['city']->content(),
    'country_id' => $country_id, 'type' => 'ezuser', 'email' => $userEmail );
// On ajoute l'adresse a un tableau d'adresses
$addresses[] = array( 0, 0, $address );
// Pour eviter des problemes si plusieurs partenaires ont le meme NOM Prenom,
// On ajoute l'id de l'objet sur EZPublish au nom de notre partenaire
$name = $contentObject->attribute('id').' :: '.$name;
// On prepare notre tableau de valeurs concernant la creation de notre partenaire
$values = array( 'name' => $name, 'active' => true, 'address' => $addresses );
// On cree le compte comptable associe à notre partenaire
$accountId = TinyERP::createAccount( $name );
// On l'ajoute a notre tableau de valeurs
if ( is_integer( $accountId ) ) {
    $values['property_account_receivable'] = $accountId;
}
// On appelle notre methode de connexion au serveur XMLRPC de TinyERP
$erpId = TinyERP::sendToTiny( 'res.partner', 'create', $values );
// Si la creation a fonctionne, nous avons recupere l'id de notre partenaire
if ( is_integer( $erpId ) ) {
    // Nous enregistrons cet id dans un attribut de l'objet de notre utilisateur
    $contentObjectDataMap['erp_id']->setAttribute( 'data_int', $erpId );
    $contentObjectDataMap['erp_id']->store();
    $contentObject->store();
    // Nous retournons l'id de notre nouveau partenaire sous TinyERP
    return $erpId;
}
return false;

```

rations notables ont été également apportées : aux modalités d'expéditions et aux frais afférents :

- ajout de nouveaux modes de transport,
- prise en compte de la taxe relative à l'éco-contribution pour tous les produits électroniques,
- gestion de colisages multiples en s'appuyant sur les fonctionnalités de Tiny ERP : à la pièce, au carton, à la palette ... en fonction des quantités commandées par le client et des conditionnements proposés par les fournisseurs,
- calcul des frais de port en fonction du poids de la commande, mais également

de ses dimensions physiques (volume, encombrement ... relatifs aux quantités commandées et au conditionnement proposé en conséquence), du statut du client (professionnel / particulier), de sa localisation géographique ...,

aux modalités de paiement :

- ajout de nouvelles possibilités : paiement par carte bancaire en 1 ou 3 fois, plan de financement, virement bancaire, par encours pour les clients professionnels en compte. Les divers modalités de règlements seront gérées différemment à la fois dans eZ publish et dans Tiny ERP : des marqueurs sur le nom des commandes seront générés à la volée de telle sorte de pouvoir facilement identifier chacun d'entre eux pour le traitement informatique (*workflow* adapté), et en extraire aisément des statistiques de ventes selon les différentes méthodes. Les encours clients seront plafonnés et vérifiés en temps réel au moment de la validation de la commande,
- paiement multi devises : euro / dollar,

aux retours d'informations relatives clients : outre le statut des commandes (en attente paiement, en traitement ...), les encours des clients professionnels seront automatiquement remontés au sein des comptes clients. De plus, une liste des installateurs de la région de résidence de chaque client, agréés par Prodigg et évalués qualitativement par les clients ayant eu recours à leurs services, sera disponible au sein de son espace personnel,

à la gestion des remises et codes promotion qui ne s'appuiera uniquement désormais que sur le module de price lists proposé par Tiny ERP. Pour chaque commande, eZ publish proposera au client le meilleur prix possible. Pour ce faire, il s'appuie sur le résultat du calcul combinatoire effectué par Tiny ERP tenant compte :

- des quantités commandées (système de prix dégressif) ou du délai restant avant arrivée de la marchandise en France (vente éclair à prix cassés),
- des codes promotions génériques (attribués à un produit donné, une gamme ou une typologie de clients spécifiques : grossistes, apporteurs d'affaires et revendeurs) et des remises personnalisées (*bons clients*).

La nouveauté du développement du connecteur entre eZ publish et Tiny ERP proposé par *Internethic à Prodigg*, par rapport au connecteur déjà existant, s'avère donc porter principalement sur la gestion de stocks multiples, le webshop d'eZ publish ne connaissant qu'un

Listing 5. la fonction createAccount

```

/* fonction createAccount
 * $name contient le nom du partenaire sous la forme 'id :: NOM Prénom'
 */
// On recupere la configuration contenue dans le fichier tinyerp.ini
$TinyERPIni =& eZINI::instance('tinyerp.ini');
// On s'interesse a la valeur definie par customerAccountId dans le groupe [Account]
// Il s'agit de l'id du compte 411, auquel nous allons rattacher les comptes clients
$customerAccountId = (int)$TinyERPIni->variable( 'Account', 'customerAccountId' );
// On separe le nom du partenaire et son id eZpublish a l'aide du delimitateur ::
$key = strpos( $name, '::' );
$partnerName = ltrim( substr( $name, $key + 2 ) );
// On cree un tableau de valeurs associant chaque lettre de l'alphabet a un nombre
$alpha = array( "a" => "01", "b" => "02", "c" => "03", "d" => "04", "e" => "05", "f" =>
    => "06", "g" => "07", "h" => "08", "i" => "09", "j" => "10", "k" => "11", "l" =>
    "12", "m" => "13", "n" => "14", "o" => "15", "p" => "16", "q" => "17", "r" =>
    "18", "s" => "19", "t" => "20", "u" => "21", "v" => "22", "w" => "23", "x" =>
    "24", "y" => "25", "z" => "26" );
// On incremente les numeros de compte en partant de 000000
$base = "000000";
// On recupere le chiffre correspondant a la premiere lettre du nom du partenaire
$placeInAlpha = $alpha[ strtolower( $partnerName[0] ) ];
// On cherche s'il existe deja des comptes comptables pour cette lettre
$values = array( array( "code", "ilike", "411".$placeInAlpha ) );
$accounts = TinyERP::sendToTiny( 'account.account', 'search', $values );
// Si il existe des comptes pour cette lettre
if( is_array( $accounts ) ) {
    // On prend le dernier indice du tableau
    $values = array( array_pop( $accounts ) );
    // On recupere le dernier compte a partir de son indice
    $account = TinyERP::sendToTiny( 'account.account', 'read', $values );
    // On recupere son numero de compte comptable et on l'incremente
    $code = intval( $account[0]['code'] ) + 1;
    // Sinon il s'agit du premier compte pour cette lettre
} else {
    // numero du compte : 411, le chiffre associe a la lettre, et 000000
    $code = "411".$placeInAlpha.$base;
}
// On prepare le tableau de valeur pour creer notre compte comptable
$values = array( 'name' => $partnerName, // le nom du partenaire
    'company_id' => 1, // l'identifiant de la societe vendeuse
    'type' => 'receivable', // il s'agit d'un compte a recevoir
    'code' => $code, // le numero du compte comptable
    'parent_id' => array( array( 6, 0, array( $customerAccountId ) ) )
    // l'id du compte comptable 411 );
// On cree le compte comptable dans TinyERP
$accountCreated = TinyERP::sendToTiny( 'account.account', 'create', $values );
// Si la creation a fonctionne
if ( is_integer( $accountCreated ) ) {
    // On retourne l'id du compte comptable cree.
    return $accountCreated;
}
return false;

```

Sur Internet

- <http://www.ez.no> – Site officiel de la société norvégienne eZ systems, éditeur d'eZpublish
- <http://www.internethic.com/solutions/ezpublish> – Pour davantage d'informations en français sur les fonctionnalités offertes par la solution eZ publish
- <http://www.tinyerp.com> – Site officiel de la société belge Tiny Sprl, éditeur de Tiny ERP
- http://www.internethic.com/solutions/tiny_erp – Pour davantage d'informations en français sur les fonctionnalités offertes par la solution Tiny ERP
- <http://fr.wikipedia.org/wiki/XML-RPC> – Définition complète du protocole XML-RPC
- <http://www.pokershop.fr> – Site e-commerce de la société Impoker, premier client utilisant le connecteur eZpublish-Tiny ERP v1
- http://www.ez.no/community/contribs/applications/TinyERP_connector__1 – Pour télécharger la v1 du connecteur eZpublish-Tiny ERP
- <http://www.prodigg.com> – Site e-commerce de la société Prodigg, premier client utilisant le connecteur eZpublish-Tiny ERP v2

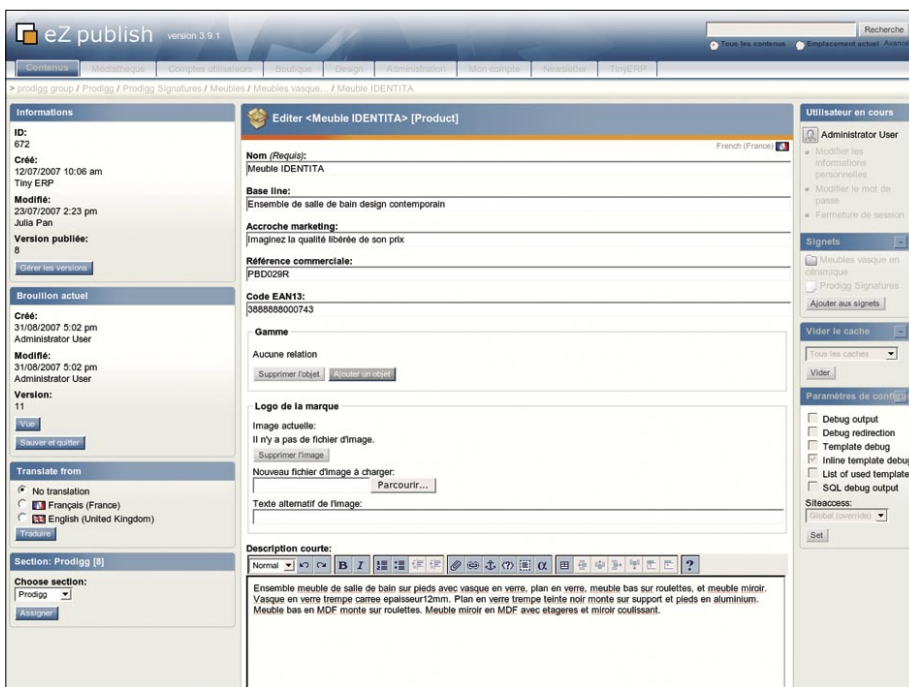


Figure 2. xxxxxxxxxxxxxxxxxxxxxxxxxxxx

seul stock : celui calculé en temps réel par Tiny ERP. Ce connecteur est par conséquent pleinement bidirectionnel et permet de profiter intégralement des possibilités de chacune des deux solutions : d'un côté la gestion de contenu éditorial, la vente et le suivi en ligne des commandes avec eZ publish, et de l'autre, tout ce qui a trait aux opérations logistiques et financières, ainsi qu'au management de la relation client (gestion des contacts, tickets et notes, devis, mailings ...) avec Tiny ERP.

Tiny ERP centralise désormais les bases de données relatives aux clients et aux produits et renvoie ses informations vers eZ publish en mode connecté : la création et la mise à jour des fiches produit s'effectue dans Tiny ERP (dénomination, référence, code EAN, fournisseur, prix, quantités disponibles ...) à l'exception du contenu éditorial (descriptif détaillé, photos, animations ...), disponible dans eZ publish.

Un workflow de validation permet de ne publier que les produits dont l'ensemble des informations descriptives ont été renseignées :

- Les fiches produit sont créées dans Tiny ERP par le service sourcing qui renseigne toutes les conditions d'achat (informations fournisseur, coûts, délai d'approvisionnement, conditionnement ...).
- Un numéro de référence unique est automatiquement assigné à chaque produit par Tiny ERP en respect de la normalisation EAN13.
- Le service marketing complète les fiches produit dans Tiny ERP avec les dénominations commerciales, les argumentaires de ventes et applique une stratégie de prix via l'assignation de *price lists* qui peuvent être propres à un client ou à un groupe de clients.
- Les fiches produits sont validées au sein de Tiny ERP avant d'être exportées automatiquement vers le catalogue produit d'eZ publish.
- L'équipe en charge du site e-commerce complète, au niveau d'eZ publish, le contenu éditorial des fiches produit (photos des produits, descriptifs détaillés, options,

produits associés, accessoires ...) avant de valider leur publication en ligne.

La gestion des stocks est confiée exclusivement à Tiny ERP :

- Les stocks sont tenus à jour dans Tiny ERP, et les délais de livraison calculés en temps réel en fonction du statut de chaque produit dans la chaîne logistique. Les produits peuvent être localisés dans des stocks réels ou dans différents stocks virtuels (en cours de transit dans la chaîne logistique) : en cours de fabrication usine, en stock départ usine (dépôt étranger), en cours de transport ... Les produits sont affectés à des unités logistiques individuellement répertoriées et des jalons de temps sont introduits permettant de disposer de différents stocks virtuels en cours de fret, et par là même d'afficher des délais de livraison les plus proches de la réalité, ainsi que le delta différentiel entre prévision et date réelle de livraison.
- Lors d'une commande passée en ligne, eZ publish s'assure en temps réel au cours de la cinématique d'achat de la disponibilité des produits commandés, en interrogeant Tiny ERP en mode connecté.
- Les stocks sont réservés en ligne dans Tiny ERP dès validation du panier puis décrémentés après confirmation du paiement.
- Un process de *workflow* d'annulation de commandes ou de délai de paiement dépassé sera développé afin que les produits des commandes annulées soient remis dans le circuit logistique, au sein du stock virtuel de destination.

Conclusion

Cette nouvelle solution se montrera beaucoup plus souple que celle précédemment développée, Tiny ERP devenant maître d'eZ publish pour tout ce qui a trait à la gestion des produits (stocks et prix), et eZ publish servant principalement de front end pour afficher ces informations.

La nouvelle version du connecteur, rendant compatible eZ publish 3.9 et Tiny ERP 4.1, sera délivrée sous licence GPL dans le courant de l'année 2008.

YANN AUTISSIER, FABRICE PEREZ

Yann Autissier, est issu de l'Ecole Centrale Marseille. Il est directeur technique chez Internethic, architecte en systèmes d'informations open sources.

Fabrice Perez, développeur certifié eZ publish, est chef de projet technique chez Internethic. Il est responsable du développement des connecteurs v1 et v2 entre eZpublish et Tiny ERP. Internethic est une SSSL marseillaise, référencée parmi les 10 premiers partenaires mondiaux d'eZ systems et contributeur officiel de la solution eZ publish depuis 2002.