# Automatisez la production de vos documents dans Tiny ERP avec OpenOffice

Fabien Pinckaers

### Sur l'auteur :

Fabien Pinckaers, 26 ans, Belgique. Directeur de la société Tiny.be, édtitrice de logiciels libres. Principal développeur et fondateur de nombreux projets libres : Tiny ERP, TeCommerce, OpenReport, Tiny Report, Tiny Raytracing, Tiny Project, Tiny Accounting, Gérant du magasin Linux en ligne : http://Openstuff.net

Pour contacter l'auteur : fp@tiny.be Bien qu'il existe de nombreux logiciels libres pour les infrastructures réseau et la bureautique, ces derniers se montrent quasiment absents du domaine des progiciels de gestion intégrés. Jusqu'il y a peu, les entreprises devaient encore se tourner vers des solutions propriétaires extrêmement coûteuses pour leur gestion avancée.

e premier novembre est sorti la version 3.1.1 de Tiny ERP, et avec elle un tout nouveau système de rapports intégrés à OpenOffice, que nous décrirons dans cet article.

Tiny ERP est un progiciel de gestion intégré libre pour petites et moyennes entreprises. Il intègre la plupart des processus d'affaire : ventes, gestion des stocks, logistique, comptabilité, gestion de projets, finance, services après ventes, chaîne de production, ...

Tiny ERP (Entreprise Ressource Planning) est caractérisé par une très forte flexibilité lui permettant de s'adapter aux besoins des différentes entreprises. Dans cet article, nous allons analyser le système permettant de créer des rapports sur mesure dans Tiny ERP.

Le moteur de génération de rapports de Tiny ERP s'appuie sur trois logiciels ;

- OpenOffice ; pour la création des templates des documents,
- La base de données objet de Tiny ERP pour l'extraction des données,
- Le langage Python pour les expressions spécifiques.

Dans cet article, nous allons adapter le format des factures au logo de notre entreprise test et nous allons créer un nouveau rapport qui affiche la liste des clients sélectionnés.

L'installation de Tiny ERP ainsi que les manuels utilisateurs et administrateurs sont disponibles sur le site: http:// tinyerp.org.

### Introduction à la génération des rapports

Il existe trois méthodes pour générer des rapports dans Tiny ERP;

- Les rapports OpenOffice
- Les rapports XML et XSL:RML

Tiny ERP		000
Eichier Uti	lisateur Eormulaire Qptions Plugins Aide	
🗅 🖬 🕯	🖹   😋 🗢 🖨   😹 📇 🛠   🖺   🗙	
Menu Sale	Order	
Order Line	Order Description : Test Order	Shipped : 🔲 Invoiced : 🕅
Payment Notes	Date Ordered : 12 05 2005 🕀	Shop : Default Shop 🗋 🗁
Links	Partner : Tiny sprl 🗋 🖻	Shipping Policy : Shipping & Manual Invoice 🗸 🗸
	Contact Address : 🛛 ien Pinckaers, 1300 Limal 🗋 🔊	Pricelist : Default Pricelist 🗋 🗁
	Invoice Address : 🛛 ien Pinckaers, 1300 Limal 🗋 😁	Project :
	Shipping Address : ien Pinckaers, 1300 Limal 🗋 😁	
	Nouveau 🔐 Supprimer 📝 Edit	
	Description Product Quantity Unit Price	Date Promised
	Debian T-shirt Debian T-shirt - M 5 12.00000	0 2005-05-12
	Debian T-shirt Debian T-shirt - XXL 3 12.00000	0 2005-05-12
		<b>v</b>
	Untaxed Amount : 96.0000	000 🔄 Total : 108.6000 💱 Compute
	Order State : Draft	Confirm Order Cancel Order
Enregistren	nent: 2 / 2 5	State: Document en cours d'édition: 2
N Nouveau	S Sauver D Effacer F Recherche X Suiva Administra	itor Requêtes: Pas de requête 🕥 🗋

Figure 1. Le bon de commande dans Tiny ERP

 Les rapports colonnes créés directement dans le client Tiny ERP.

Nous nous intéresserons à la première méthode qui est à la fois la plus simple et la plus flexible.

Le processus de création d'un nouveau rapport dans Tiny ERP est le suivant :

- Conception du rapport dans Open Office avec des expressions pour la base de données
- Génération du fichier intermédiaire .RML
- Inscription de ce fichier dans Tiny ERP

### **Principe**

Le production des documents dans Tiny ERP (factures, notes d'envoi, bons de commandes, ...) se fait grâce à des fichiers RML. RML est un format de fichier XML décrit par Reportlab [*http:* //reportlab.com/docs/RML\_UserGuide\_ 1\_0.pdf]

Le RML étant un dialecte du XML, ces fichiers sont plus simples à manipuler que directement des PDF. Ils sont ensuite convertis en PDF grâce au programme tiny\_ rml2pdf du projet *http://OpenReport.org*.

Ces fichiers RML décrivent la structure d'un document ainsi que les expressions et les champs de la base de données à y inclure. Exemple de fichier RML. <!DOCTYPE document SYSTEM "rml\_1\_0.dtd">
<document filename="example\_2.pdf">
<template>

cpageTemplate id="main">
 <frame id="first" x1="2cm"
 y1="2cm" width="17cm"
 height="26cm"/>
</pageTemplate>

- </template>
- <stylesheet/>
- <story>

<para>

Les textes <b>RML</b> sont placés dans les tags "story" qui composent le corps du document.

</para> <para>

Les paragraphes sont placés entre les tags "para" et "/para".

</para> </story> </document>

Pour l'incorporation des données provenant de la base de données de Tiny ERP dans le document, on y ajoute des expressions entre double crochets.

Ces expressions sont au format du langage de programmation Python et les variables seront remplacées par les valeurs de la base de données. Seulement, écrire à la main ce genre de fichiers .RML n'est pas toujours simple. On utilise alors OpenOffice pour construire un template de document que l'on sauve en .SXW. Ce fichier .SXW est alors converti en .RML grâce au programme : tiny\_sxw2rml.py, du projet http:// TinyReport.org.

La chaîne complète de transformation du document est présentée sous la Figure 2. Le seul fichier à créer pour construire des nouveaux rapports dynamiques dans Tiny ERP est le fichier Open Office (.sxw) contenant les expressions entre double crochets.

## Modification des factures

La première étape consiste à de construire le template du rapport dans Open Office.

Pour notre premier document, nous allons modifier un rapport existant dans Tiny ERP; l'impression des factures.

Ouvrez le fichier 'invoice.sxw' qui se trouve sur le server Tiny ERP à l'adresse : server/bin/addons/account/ report/invoice.sxw. Ce document est le template d'une facture Tiny ERP.

Pour notre premier essai, nous ne ferons que des modifications simples, sans



**Figure 2.** Le processus de transformation du document vers le PDF

💿 in	voice	.sxw - O	penOffice.org 1.1.4	111					0	00
<u>F</u> ile	<u>E</u> dit	<u>V</u> iew <u>I</u>	nsert F <u>o</u> rmat <u>T</u> ools	Window Hel	р					×
	6		2 🔂 🚨 🕺	- E (	) () (	> 🖪 (	) 🖾	75% 💌		
Table	Con	tents	▼ Luxi Serif	• 12 •						<b>•</b>
	L	· · · 1 · ·	×···1;··2·;·3··;4·	··5···6···7·	7.8. <sup>1.</sup> 8.7	10 • 11 •	12 • 13 • 1	4 • 15 • 16 • 1	7 • 18 • •	
0			Tinv E	RP	Tel: (+32).1 Mail: fp@	0.68.94.39 Ptiny.be	Te fdsfs	csas sprl dsfdsfdfsd		
ø			Open Source Busines	s Solutions	Web: http://ti	inyerp.com	B-145	57 Walhain		
							Walh SALE Of	ain, 2005-09-19 RDER INVOICE		
ABC			[[ objects.repeatIn('o	0)]]	(17 H o.)	/		[[ 3 - 12 ]]		
I			INV	OICE [[ '%	60/ <u>a</u> ' %	1111( <b>0</b> .111	imber) []			
ABC					[	o.address_	invoice_id.na	me ]]		
RBC					[	[ o.address_	invoice_id.str	eet ]]		
					[	[ o.address_ [ o.address_	invoice_id.zi p invoice_id.ci t	9]] Y]]		
					C	[ o.address_	invoice_id.com	untry_id.name ]]		
9					Π	[VA: [[ 0.pa	rtner_id.vət ]]			
9										
2			Product		Taxes	Quantity	Unit Price	Subtotal		
			[[o.invoice_line.name]]		[[ o.invoice _line.invoic e_line_tax_i d.name ]]	[[o.invoic e_line.qua ntity]]	[[ o.invoice_ line.price_u nit ]]	[[ o.invoice_lin e.price_subtotal ]]		
				SUBTOTAL:				[[ o.amount_unt axed ]]		<b>▼</b>
			[[ o.tax_line.name ]]					[[ o.tax_line.am ount ]]		40
	4									▶
Page	21/1	L	Default	75% IN	SRT STD	HYP *		A1		

Figure 3. Le template de facture dans OpenOffice

toucher aux expressions entre [[]]. Par exemple, vous pouvez changer le logo, les coordonnées de votre entreprise, les couleurs de fonds, ...

Ensuite, il faut pouvoir importer le document généré dans Tiny ERP. Pour ce faire nous utiliserons le programme tiny\_ sxw2rml. Vous le trouverez à l'adresse : http://tinyreport.org, section download.

La facture OpenOffice contenant les expressions.

Convertissez alors votre fichier :

### ./tiny\_sxw2rml.py path\_to/invoice.sxw >path\_to/invoice.rml

Lorsque vous imprimerez des factures dans Tiny ERP, celles-ci seront adaptées aux modifications apportées.

#### Les expressions

Les expressions entre double crochets "[[ ... ]]", sont des expressions au format du langage Python qui font appel à la base de données. Les champs de la base de données sont accédés comme de simples variables. Les relations d'une table à l'autre sont accessibles avec un point ".". Exemple, supposons que l'on fait un document pour une facture, possédant les champs suivants (décrits au format SQL):

- [[ 3\*2+4 ]] sera remplacé par 10.
- [[ name ]] sera remplacé par le champ 'name' de la facture

- [[ len(name) ]] sera remplacé par la taille du champ name de la facture (len est une fonction Python)
- [[ o.partner\_id.name ]] sera remplacé par le champ 'name' du partenaire référencé par le champ partner\_id de la facture.
- [[ o.line\_ids.name ]] la ligne du tableau dans lequel se trouve cette expression sera dupliquée pour chaque ligne de facture et dans cette colonne, la valeur du champ « name » de chaque ligne sera ajoutée. La ligne est dupliquée car le champ line\_ids de la facture est une relation de cardinalité one2many.

Ainsi, lorsque vous utilisez les champs one2many de Tiny ERP (par exemple, une facture qui possède plusieurs lignes), le moteur de Tiny ERP va dupliquer la partie du rapport dans lequel se trouve cette expression. L'effet dépend de l'emplacement de cette expression :

- dans un tableau: duplique la ligne,
- dans une section OpenOffice : duplique la section,
- ni dans un tableau, ni dans une section : duplique la page en cous.

La seule variable accessible au départ est : "objects". Elle contient la liste des objets sélectionnés dans Tiny ERP lorsque l'utilisateur a cliqué sur l'icône "Print".

Vous pouvez créer d'autres variables grâce à la méthode "repeatIn('variable\_

💿 partner_list.sxw - OpenOffice.org 1.1.4 🥼		000
<u>File Edit View Insert Format Tools Window H</u> elp		×
B 🕞 🕞 🕑 🔂 🖴 🗴 🗅 🎁 🚿 🖉 🍕	> 🌇 🔊 🖾 100% 💌	
Table Heading 🔽 Luxi Serif 🔽 12 💌 🗛 🥂		
	· · 8 · 2 · 9 · · · 10 · · · 11 · · · 12	·13· · ·14· · ·15· · ·16· ·
	Tiny sprl	
	Rue du vieux Ch	ateau 21
	B-1457 Walk	ain
R ► <del>V</del> RBC		
Nombre de clients	: ([[ len(objects) ]])	
-		
Nom du client	Website	Lang
[[ objects.name ]]	[[ objects.website ]]	[[ objects. lang ]]
		<b>•</b>
		<u>*</u>
		¥
		<u>&gt;</u>
Page 1 / 1   Detault   100%   I	NSRT   STD  HYP  *	BI

Figure 4. Le template de listing client dans OpenOffice

### Tiny ERP

Tiny ER	Rue du vieux B-1457 V re de clients: (5)	sprl Chateau 21 Valhain	
Nom du client	Website	Lang	
Agrolait	http://agrolait.be		
Banque Plein-Aux-As			
China Export	http://chinaexport.cn	en	
Ditrib PC			
ASUS	http://asus.com	en	

Figure 5. Le listing produit en PDF depuis Tiny ERP

name')". Ainsi, l'expression : "objects. repeatln('o')" dupliquera la ligne ou la section en cours pour chaque objet et vous pourrez utilisez la variable "o" pour accéder à chaque objet indépendamment.

### Création d'un nouveau rapport

Dans les précédents rapports, nous avons modifié un rapport existant dans Tiny ERP. Nous allons maintenant créer notre propre rapport. Pour ce faire, les étapes sont similaires, si ce n'est qu'il faut également inscrire le rapport dans le serveur Tiny ERP.

Créez le fichier suivant dans Open-Office : et l'on sauve ce fichier dans le répertoire du serveur Tiny ERP : "server/ addons/base/res/partner/report/partner\_ list.sxw".

C'est un rapport sur une liste de clients (appelés partenaires) qui affiche un tableau avec le site Internet et la langue de chaque client.

Il faut ensuite transformer ce rapport en RML à l'aide de tiny\_sxw2rml.py : *tiny\_ sxw2rml path\_to/partner\_list.sxw, path\_to/ partner\_list.rml* (remplacez path\_to par le chemin de votre installation Tiny ERP).



Figure 6. L'édition des données de l'utilisateur

La dernière étape est d'inscrire ce rapport dans un des modules de Tiny ERP, dans ce cas-ci : "base". Chaque module possède une liste de fichiers .XML qui seront chargés à la mise à jour du programme, cette liste peut être obtenue dans le fichier : *tinyerp/server/addons/ base/\_terp\_\_.py*.

Dans notre exemple, nous allons inscrire le rapport dans le fichier : *server/ addons/base/res/partner/partner\_ report.xml.* 

Et nous ajoutons l'inscription suivante :

<re< th=""><th>port</th><th></th><th></th></re<>	port		
	string="Partner	List	Test
	model="res.parts	ner"	
	name="partner.l:	ist"	

rml="base/res/partner/report/
partner list.rml" />

Où :

- string: désigne l'intitulé du rapport (en Anglais, sera traduit en français après par le système de traduction de Tiny ERP).
- model : désigne l'objet sur lequel se fait le rapport
- name : nom du rapport, au choix.
- rml : chemin du fichier RML que l'on vient de générer.

Pour que Tiny ERP prenne en compte ce nouveau rapport, il faut l'exécuter avec l'argument --update=base : tinyerp\_ server.py --update=base.

Ensuite, dans le client Tiny ERP, on peut faire une sélection de partenaires et l'imprimer. Cela donne le PDF représenté sous la Figure 5.

### Un moteur de rapport indépendant

L'équipe de développement de Tiny ERP travaille sur un projet permettant d'utiliser ce système de rapport indépendamment de Tiny ERP.

Ce projet est toujours en cours de développement mais vous pouvez suivre son évolution sur *http://tinyreport.org*.

Cela permettra de générer des fichiers PDF dynamiquement pour vos applications et vos sites webs sans devoir écrire une seule ligne de programmation.

o test2.sxv <u>F</u> ichier <u>É</u> dit	w - OpenOffice.org 1.1.4 (() tion <u>Affichage I</u> nsertion Forma <u>t O</u> utils I	Fe <u>n</u> être A	i <u>d</u> e		<u>000</u> ×
	<b>.   🖉 🖓 😃  </b> 🗶 🛅 🗳	@   4	> 🔛 🕥	75%	•
Standard	💌 Luxi Serif 💌 12 💌				A V
	<u></u>	9··· 10·· 11	• 12 • 13 •	14 · 15 · 16 · 1	ζ··18·· 🔺
0	Test Rep	port			
	Partners Contacts (	[[ len(obj	ects) ]])		
	Partners Contacts () [[ objects.address.repearIn('o') ]]	[[ len(obj	ects) ]])		
	Partners Contacts () [[ objects.address.repeatIn('o') ]]	[[ len(obj id.name ]] –	ects) ]])		
	Partners Contacts () [[objects.address.repeatIn('o') ]]	[[ len(obj id.name ]] ner_id.credit	iects) ]]) - ]]	CITY	
	Partners Contacts () [[ objects.address.repeatIn(o') ]]	id.name]] – ner_id.credit ZIP	iects) ]]) - ]] <u>Email</u> [f o email ]]	CITY [[acity]]	
A C	Partners Contacts () [[objects.address.repeatIn(o)]]	id.name ]] id.radit id.credit ZIP [[ 0.zip ]]	iects) ]]) ]] [[ o.email ]]	<u>CITY</u> [[o.city]]	
	Partners Contacts () [[objects.address.repeatIn(o')]]	[[ len(obj id.name ]] — ner_id.credit ZIP [[ o.zip ]]	ects) ]]) ]] [[ o.email ]]	<u>CITY</u> [[o.city]]	
	Partners Contacts () [[objects.address.repeatIn(o')]]	[[ len(obj id.name ]] — ner_id.credit <i>ZIP</i> [[ o.zip ]]	iects) ]]) ]] <u>Email</u> [[ o.email ]]	<u>СІТУ</u> [[ o.city ]]	►

Figure 7. Le rapport du test dans OpenOffice