SOLUTION REFERENCE:	Comp/POS/NPNG/5May/Red1/1.0
APPLICATION:	Compiere ERP/CRM v2.5.0e on Oracle 9i
EXTENSION:	Point Of Sales System
VERSION:	1.0
DATE:	May, 2004
AUTHOR:	red1 OPEN fiX Sdn. Bhd., Malaysia
EMAIL CONTACT	redhuanoon@yahoo.com
CLIENT:	No Pain No Gain Sports Centre, MK Holdings Group

I am using the Queen's English. Where you find it to be broken, its probably from some glass jar.

CONTENT

OPENfiX is dedicated to implement Open Source Financial software excellently. The OPEN stands for Open Source Software. The ‘fi’ emphasizes Financial Systems. The term ‘fix’ is self-explanatory.
Other trademark benefactors in these documents are **Compiere**, **Sun MicroSystems** for **Java**, **IBM** for **Eclipse, Microsoft** and **Oracle**.

Please view this document in WindowsXP Microsoft Word for accurate layout.

I hereby anoint the Open Source advantage over my humble snippets. Please use them freely and any bugs you refered back to me will be kept in a glass jar for further study.

DEDICATION

There are times when no words are adequate to express gratitude other than by a more demonstrative and appropriate action. Thus I am contributing this as dedication to particular people.

Firstly this goes to Robin SP Hoo who is the Malaysian Translator for Compiere, for having introduced Compiere to us last year, and we have never looked back since. Largely so due to his constant ramblings about retiring early but maintaining a generous unselfish sharing spirit that keeps my interest and passion for Compiere alive, even though there were rough patches along the way.

I am making this contribution public at the time he is due in Bonn, Germany for the May 2004 Compiere training. Hope he can show his coursemates from there – our RM0.08 (2 cents US) worth in the Open Source world.

My 2nd dedication goes out to one of the main reasons he is there for, which is Jorg Janke and his Compiere team. For Mr. Janke has made this pioneering effort of porting a complete suite of industry class commercial value software onto Open Source. They have proven not to accept the leading edge. They are at the bleeding edge!

The most I can do is contribute back. So, please make way a little.

sounds of euphoric clapping, now fading off

HAVE A GREAT TRAINING WEEK, PEOPLE!

- advertising space -

Point of Sales (POS)

Introduction

Recently our company went bonkers when upon securing its very first deal, was asked to deliver a Point Of Sales system to a Sports Centre that houses Miniature Football Courts or Futsal and so has two POS stations – a cafeteria and a front desk counter for handling the facilities resources.

Compiere fits like a glove when it comes to such things as courts booking as the Resource Assignment portion is already there with a Calendaring System to manage the reservations and selections. I only need to code slightly to make it automatically display customer info without redundant input.

However the POS for accepting cash payments from customers needs some customisation as the present Sales Order screen is too comprehensive and cluttered and does not bear the universally required *Given Amount* and *Change Amount* fields. Also the POS needs to open an electronic cash drawer on succesfull completion of the sales! If that isn't enough, I also need to figure out how to print the receipt on a teeny meeny A7 size paper roll! And without needing to choose the default format!

All that within a record time benchmark - something like 5 seconds from punching the order to before the receipt prints!

The good news is that we achieved all that and we live to tell the story. Also at the end of the story you will find how Robin Hoo with his insight into Compiere manage to twitch the interface slightly to marvelously transform it into a sure-fire POS, if I may dramatise further.

We roll out the solution in a record time of one month inclusive of user coaching and accounts integration. We have to be pretty fast since their facilities open within one month of us going bonkers.

But there was one close call. When in the middle of our customising, the customer raises doubts as to our POS interface been rather unfriendly looking and asked us to consider purchasing a 3rd party ready-made POS instead. That call was indeed close until Robin gave the interface screen that magic twitch badly needed to save the day.

Other steps that we took to make the interface faster is to reduce the key strokes for punching in an order. That requires more common sense, than coding – I will show you why later.

I use Eclipse, an IBM Open Source project to toss the source codes. There were many sleepless nights I went through cracking at the codes, but they are mainly due to the steep climb up the learning curve. You see, I have never encountered the codes before this and I just started drinking Java!

I even manage to learn how the Callouts work along the way. Please learn this too, as Compiere Callouts are pretty nifty and lightning fast to program. I'll make sure I rub that skill onto you.

This, coupled with Compiere's framework, allowed the customer to have operations, inventory, accounting and user level control, all integrated seamlessly with real-time reporting. Also the main reason why the customer chooses us is that we gave the best value for money! Nothing else is within reach.

From Sales Order to POS

Before we do that lets look at the original screen we need to work on – the Sales Order screen.

You have to pull up the Screen yourself from Compiere as my word file is getting to big to store all the graphics.

You will see that it looks pretty cluttered. We will end up with the POS screen below. Just to give you a quick peek to keep the spirits alive.

By the way, notice the Liquid Look I am using ☺ . Blerp, blerp!

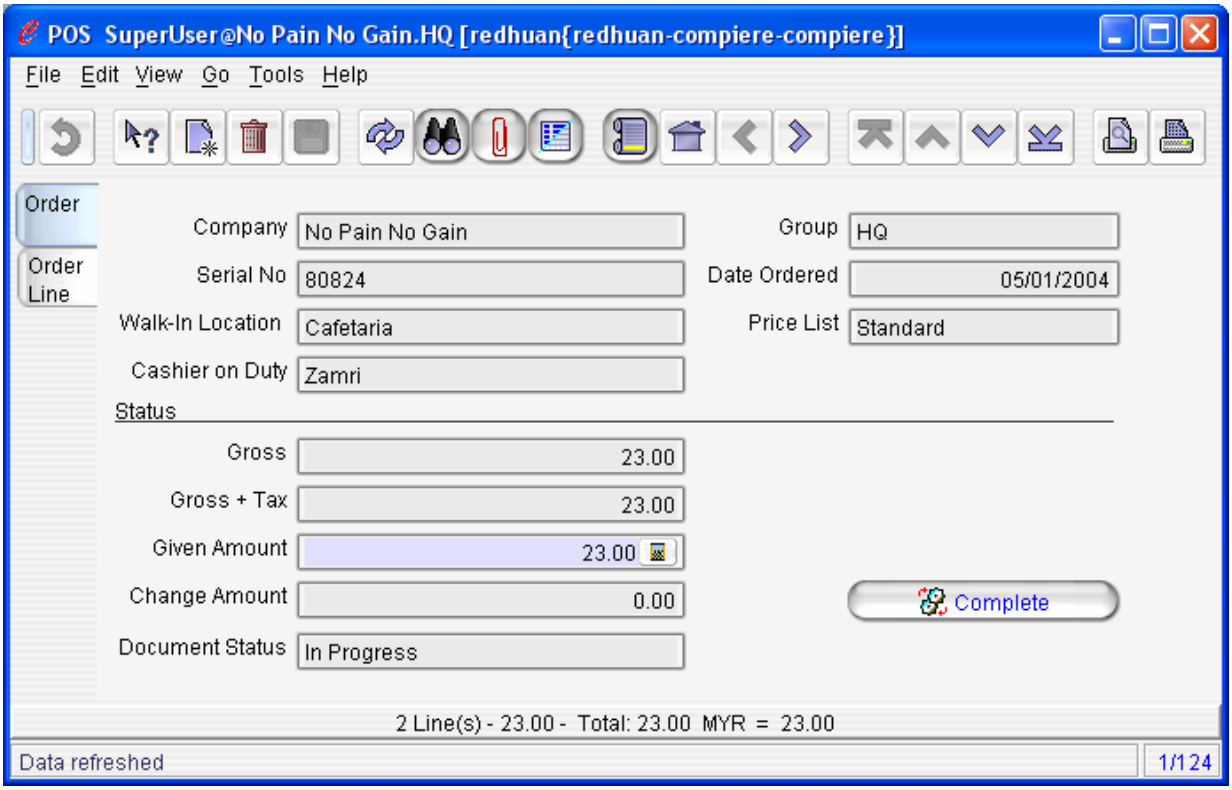

Figure 1 – POS Screen, first tab showing the Given and Change Amounts fields

The next page shows the second tab that corresponds to the original Sales Order. Notice that I removed the tax tab to avoid been too busy. These transformations, both above and below was done via the wonderful Application Dictionary, especially the multi row trick (done by Robin) which is achieved, with a few easy steps. Thus you need not do any source code changes. At least not yet.

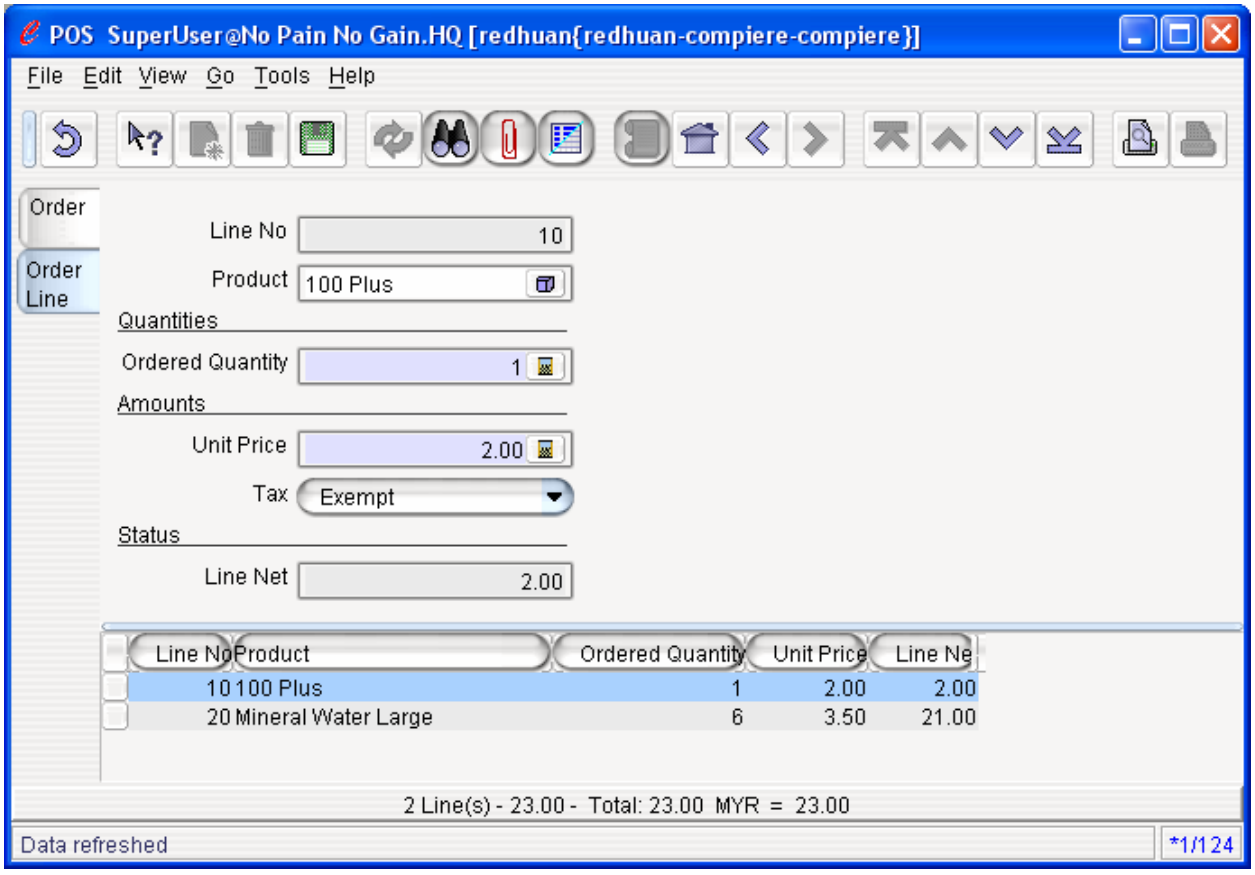

Figure 2 – POS Screen, the second tab which shows the multi row interface

I hear that Compiere is due to come up with the Master-Detail display pretty soon. So that means we can do away with twin Tabs, promising an even more craftier interface. I can't wait for such goodies round the corner. For the time being, I gave the customers a quick tab change by using function keys to jump from tab to tab. For those who are quick with combination keys, they can use the present Alt-> and Alt-< keys.

Another peek, on the right is the receipt.

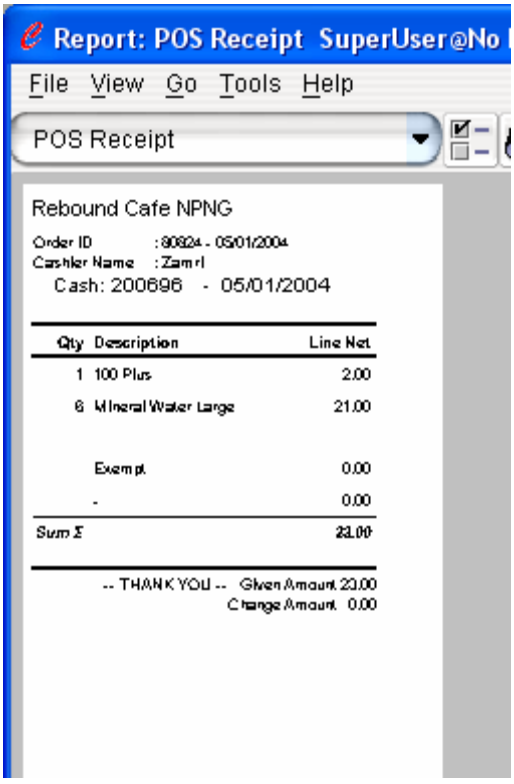

Figure 3 – POS Receipt, configure for A7 size paper

Using the Application Dictionary

Create New Window

1. Login as System Administrator and click on the Window, Tab & Field item.
2. Create a new Window, give a name, i.e. Cafeteria or POS.
3. Select Copy Window Tabs from the Sales Order window.
4. Go to the Tabs portion and notice that they are been populated exactly like the Sales Order Window. Go to the Fields portion and again notice they are populated accordingly.
5. Deactivate or uncheck the IsActive for fields that you do not need. You can use my screenshots above as a guide. You can also delete the fields right out, but only if you feel lucky. I hate to be around when you go “oh oh!”.
6. Take note of the details shown below. Copy them judiciously.

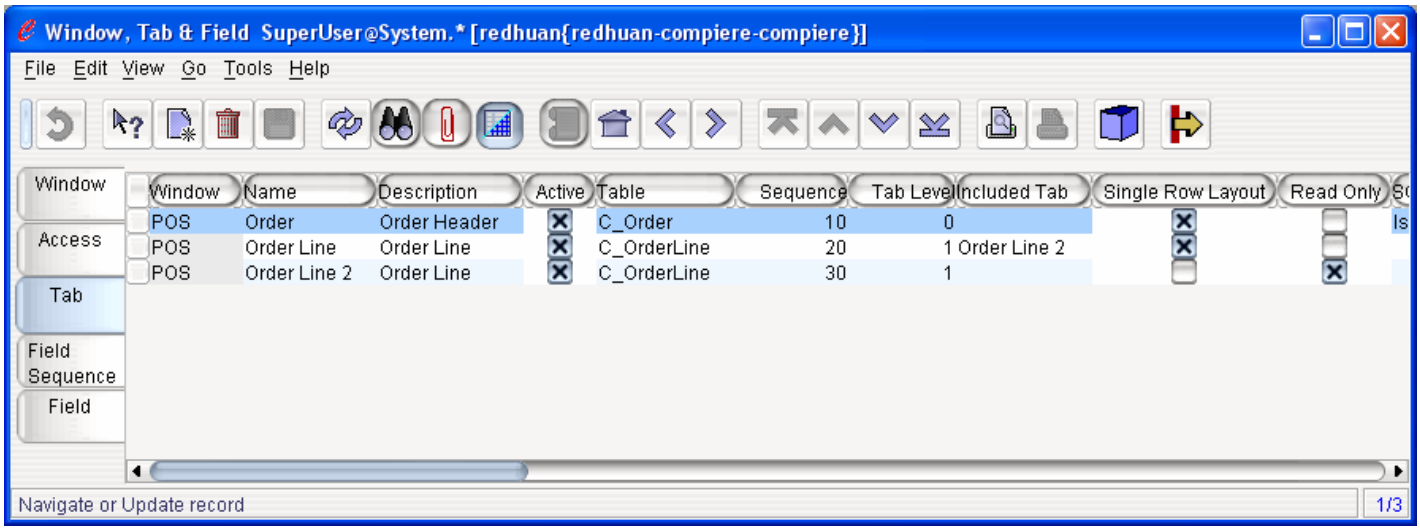

Figure 4 – AD Window configure, POS has 3 tabs, Tab 2 includes Tab 3 for the multi row view.

Keep referring to the POS Screen earlier for all these pointers.

7. Create a Standard Walk In Customer (from Business Partner menu) calling it Cafeteria.
8. Make it the preference (right click on the Cafeteria).
9. Also right-click and made POS as the preferred Document Type, but made that Not Displayed to avoid clutter.
10. Rename the Sales Representative field as Cashier-on-duty.
11. Make it defaulted to the login user. (place @#CreatedBy@ in the Default Value field of the Table and Column window for C_Order, under Sales_Rep_ID column).
12. Come back to include the two new fields after creating them in the Table and Column.

Create New Fields

1. Click on the Table and Column item from the System Menu.
2. Call up the C_Order table.
3. Add two new fields: GivenAmt, ChangeAmt. Declare them as amounts.
4. Note the Callout field at the bottom.
5. Type in a full java method name, i.e. org.compiere.CalloutSystem.GivenAmt.

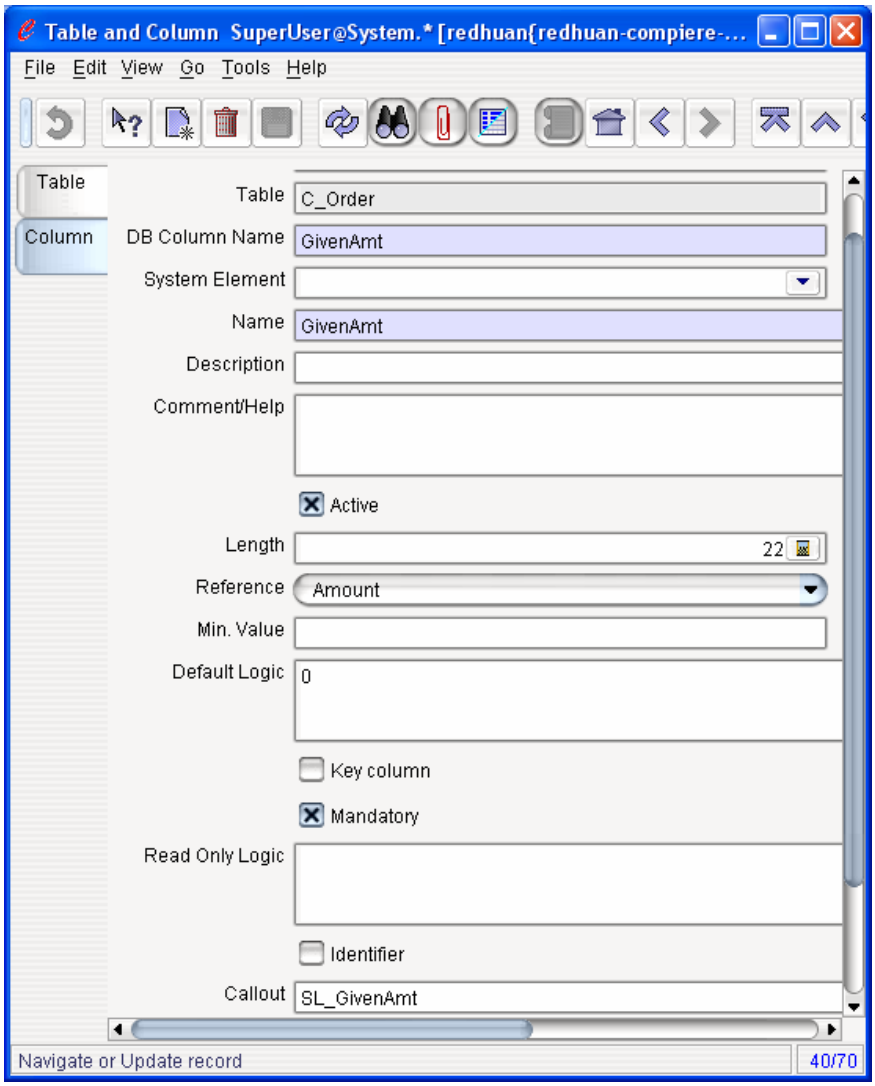

Figure 5 –GivenAmt column callout. The java method stated must exist before compiling.

When the GivenAmt is filled, a callout is activated. We will need a java method that calculate against the total price for the ChangeAmt.

You can alternatively use SL_ before the method name instead of a fully quallified one, as this will call the method in the CalloutSystem.java Next we will go and create that method. Yummy!

Development and Coding

Using the Callout

Now, are you ready to enter *The Matrix*? Then call up your Eclipse, with Compiere250e imported and let's dial in.

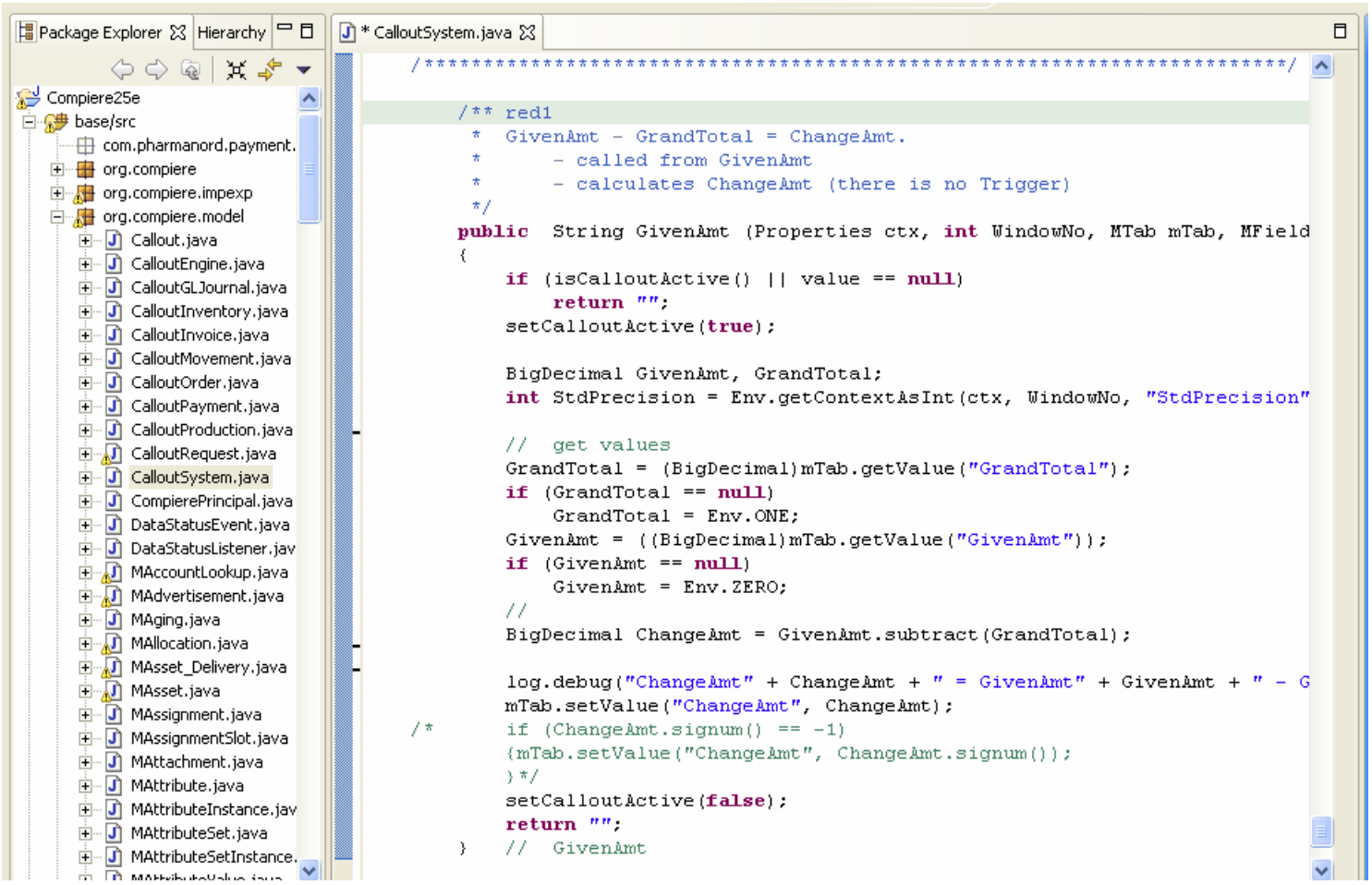

Figure 6 – Eclipse: call up org.compiere.model.CalloutSystem.java and create method *GivenAmt*.

Here's a quote on the light side – "Good Artists Copy, Great Artists Steal".

There isn't much dificulty to figure out a callout. I just looked at other methods for comprehension. Then I copied one likely suspect, rename it accordingly and replace the arithmetic function needed. Eclipse IDE auto code-assist took the trial and error work away from me.

Opening the Cash Drawer

Its common for a POS System to control the cash drawer. So we got a vendor to supply an electronic cash drawer, where the manual says that it will open when it receives a single char print from its COM Port connection. So I hook it up to my development PC and set down to crack at it. But how does those codes that does this Open Sesame looks like? Remembering that they must be in Java, and not Visual Basic which was stated in the manual. With no original programmer around to ask from, I went into Google. The problem is asking the right question. So I phrase and rephrase: 'com port java', 'native call com port', 'VB call Java', 'HELP ME SOMEONE!'.

About 5 hours later I come across that few strands of code. And here I produce them into two java files which are shown below.

Strip.java (found in the Tools package of Compiere)

```
/**
                * Prints to serial port1 for drawer to open
                * by Red1 - 2004,April19
                */
        public void OpenDrawer()

                {

                 try {
                  byte data[] =
                          " ".getBytes();
                  FileOutputStream fos = new FileOutputStream( portname );
                  BufferedOutputStream bos = new BufferedOutputStream( fos );
                  fos.write( data, 0, data.length );
                  fos.close();
                         }
                 catch( Exception e ) {
                  e.printStackTrace();
                                        }
                }
```

Figure 7 – The codes highlighted in blue are outright 'adapted' from a kind soul on the web.

Here is my own amateur moves: (in red)

VDocAction.java

```
import org.compiere.tools.*

                if (index == 3)
                        {       Strip openD= new Strip();
                                openD.OpenDrawer();
                  }
```

Figure 8 – Code to call OpenDrawer method in Strip.java (Tools package) from VdocAction,java.

So here I was attempting to encapsulate that method in a convenient place called Strip.java under the Tools package which has import of java.io, and call it OpenDrawer. I have that hooked to the spot where the trigger happens – VDocAction.java which handles the Complete button routine.

Figure 9 – Complete button to process the POS Order.

To finally pinpoint that it is this java class took me some hours to trace in Eclipse using breakpoints and stepping through the action in question.

Upon arrival at the particular code snippet, I tested that index==3 qualifies a proper complete process whereas other values are for void and reversal.

So this is smart in the sense that the drawer will only open when the cashier does a complete sale and not when there is a reversal or void, which she has to call upon her supervisor to bring the key to manually open the cash drawer. Pretty smart move, isn't it?

Getting to clean codes always builds tension. When I first tested the raw code in Eclipse, round after round without the drawer giving the slightest tick. When the drawer suddenly opens after the Nth code shuffling, I really jumped. I ran out of the room screaming, "It opens!! The cash drawer does open on its own!".

Then came a slight problem. When I build and compile the CClient.jar, it claims that there is no such OpenDrawer method. Rightly so, for I placed it in the tools package which is queued for later compilation than the package, org.compiere.grid.ed. So, I apply the School of Quick and Dirty

Technique. I remove my elegant call to Strip.java, and dump everything under one place. So much for trying to show off my amateur O-O skills! Here is the final smashed potato:

```
*     Save to Database
       *  @return true if saved to Tab
       */
      private boolean save()
      {
            int index = getSelectedIndex();
            if (index == -1)
                  return false;
//red1 - Opens the Cash Drawer
            if (index == 3)
                  {
      {
//    Runtime rt = Runtime.getRuntime();
//    Process p = null;
      String portname = "com1:";
  try {
   byte data[] =
       " ".getBytes();
   FileOutputStream fos = new FileOutputStream( portname );
   BufferedOutputStream bos = new BufferedOutputStream( fos );
   fos.write( data, 0, data.length );
   fos.close();
   }
  catch( Exception e ) {
   e.printStackTrace();
                                       }
      }
                  }
// red1 - end
            //    Save Selection
            m_mTab.setValue("DocAction", s_value[index]);
            return true;
      }     //    save
```

Figure 10 – Final codes, compiled and tested OK.

At the top of VDocAction.java you have to include an import call for java.io like this: `import java.io.*;` . The IO classes handle these traffic matters without bothering us on how they actually do it. That's the point of O-O's encapsulation or information hiding, whatever.

After compiling, the code works fine, and the drawer and I began a fetish relationship for some time. Just watching it open and pushing it back to just click shut, and doing it over and over again.

Then I had a lot to explain to the accountant for all the reversed bills. Humans just do not understand us robots.

Setting A7 Paper Size

It's a different curve when it comes to the Paper size. For those who are familiar with the Printing portion of Compiere, will notice that there is a Print Paper option to select your paper. But when called upon, there is not much choice as you can see here.

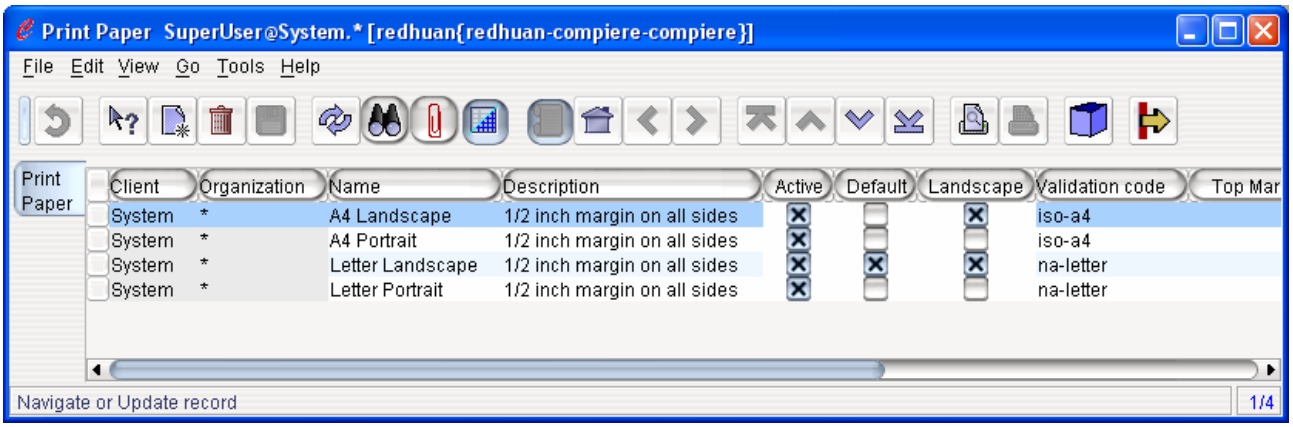

Figure 11 – Print Paper window, you have to create a new paper type and follow up with code.

I innocently thought that by adding a new record that spells out A7 will summon that paper size, but nothing of that sort happens. Note the iso validation code on the right column. That is referred in the code shown below. So, I went back to the ever faithful Eclipse IDE, to sniff out the exact code that controls the paper size and soon I arrive at the rightful source, which is in MPrintPaper.java.

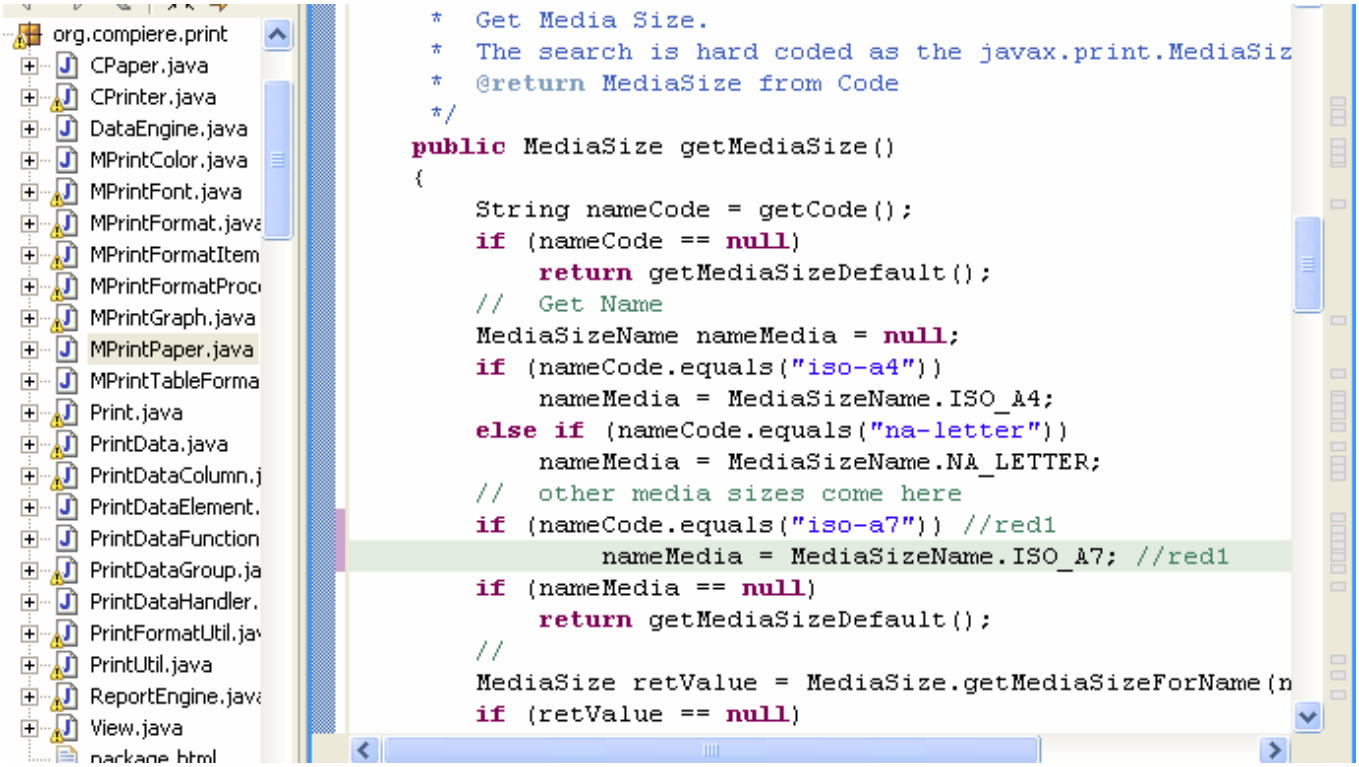

Figure 12 – Copy and amend to make new reference to ISO_A7 paper size.

As you can see in the above code portion, we are greatly assisted by the luxurious documentation that illuminates throughout the Compiere Source. It doesn't take terribly long to figure out the exact snippet to add to getMediaSize().

Below where the comments say: 'other media sizes come here' I obediently copied the a4 codes and paste anew. Hehe! I am a pretty good artist aint I? Then I remove the ISO_A4 part and just place a dot after MediaSizeName to obtain a list of its available methods (see below). Going through the list and trying a few likely candidates and print out receipts to check its layout correctness. You may choose a different one, and if you do, you can retrospect the type definition accordingly in the iso type and Print Paper table. Notice that the iso type in the screen shot below is iso-c6 which I experimented with but finally switch to a7.

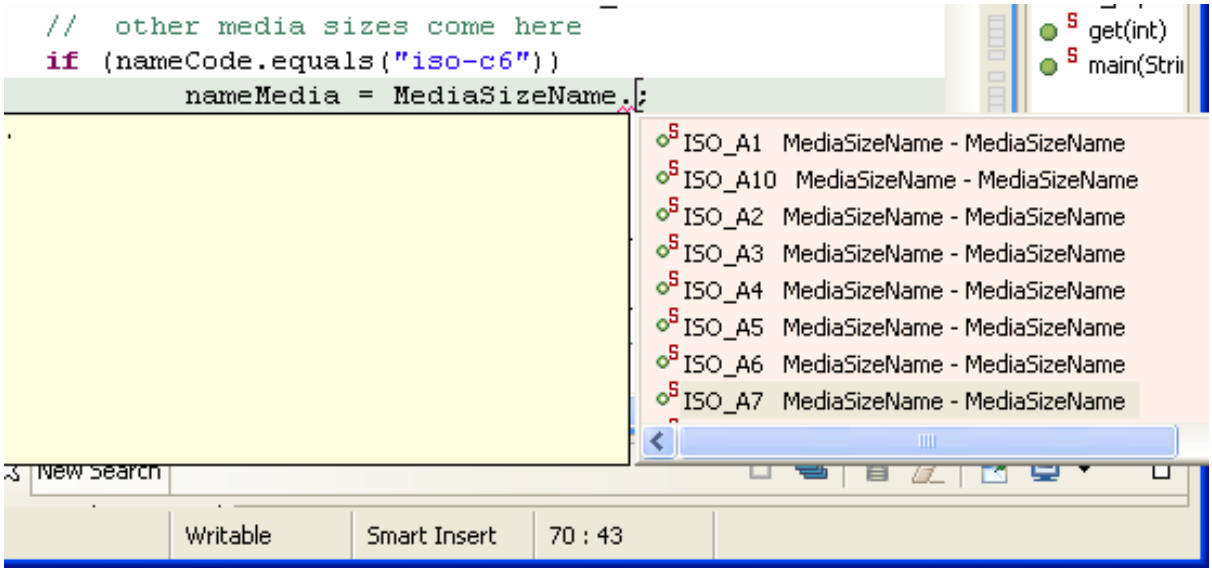

Figure 13 – All the MediaSizeNames you want.

Then we have to adjust the Print Format to fit into that tiny paper size. I turn all the fonts to the smallest possible which is size 8. Unwanted luggage is thrown out of the layout. Just remember to do a cache reset everytime you cannot see your changes in the print preview. Usually the app load from cache which may not be an updated version of your print format.

Now we set our attention to making the Print Format to default to the Receipt format when the cashier hits the print button. I also convert the function key F12 to be a one key print command. We will now look at making that default receipt print.

Setting Default Print Format

Compiere has incorporated the rule that preferred Print Format is controlled in a hierarchy fashion. If the Business Partner's Customer tab has its Print Format field selected, then that has precedent above all other settings. The next setting in line lies with the Document Type table.

Since we have set the POS Order to have only one business partner which is Cafeteria, then we set the Print Format there to be POS Receipt. (Select that from the Customer tab as shown below).

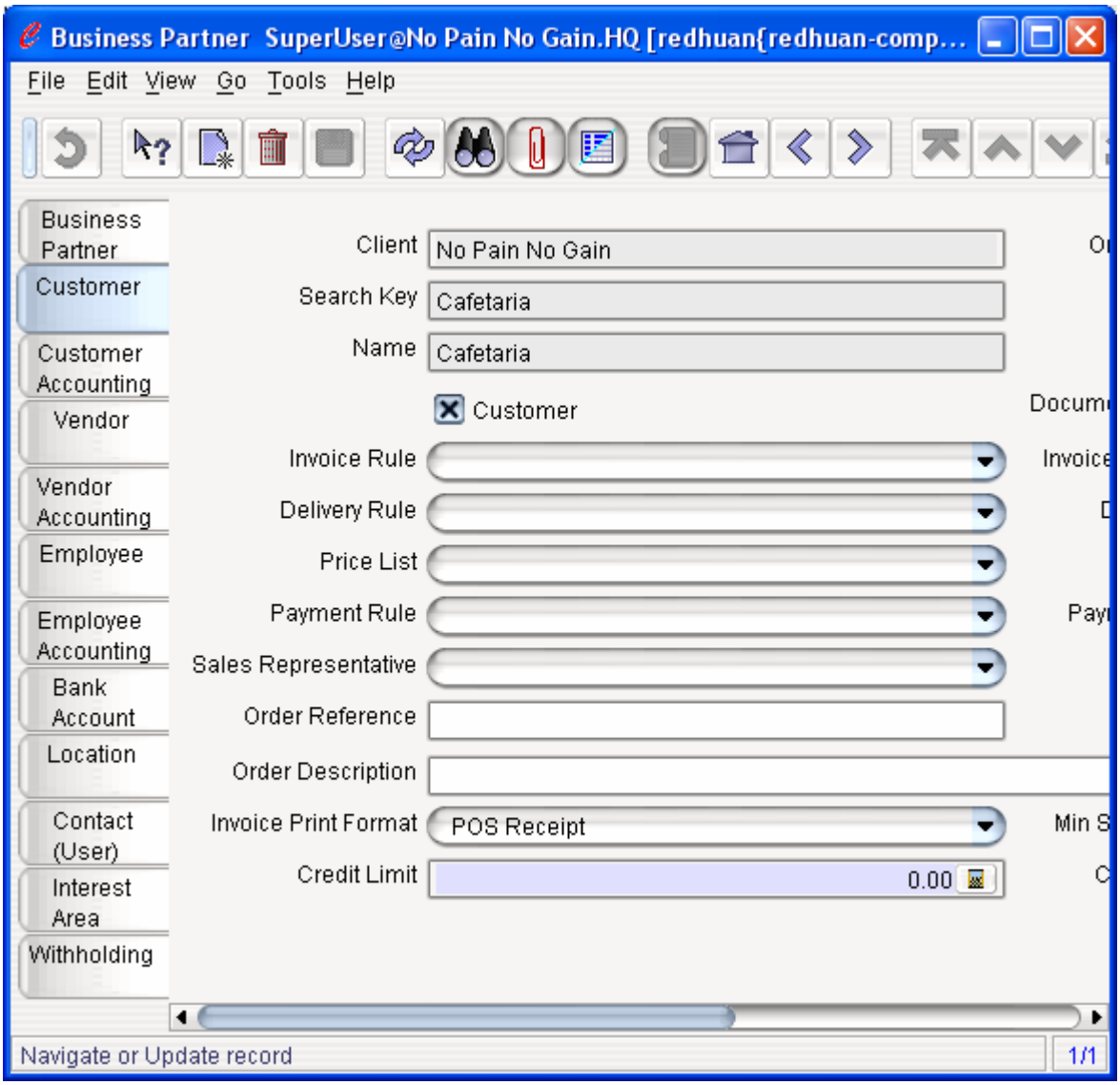

Figure 14 – This is first place a Print Format is check for default printing.

For your information when a POS Sales Order is processed, it is converted into an AR Invoice Indirect which carries an Invoice Header Template Print Format. Thus when you want to copy a print format to modify into the receipt, you copy from the Invoice format. Do not copy from a Order Template format. This also means that you cannot get a proper receipt until you complete the POS Sales process.

End Processing

Let me explain a bit more about POS sales. You probably remember that the payment term field was orignally in the Sales Order but was made not displayed in our POS Screen to avoid clutter. The Payment Term defaults to cash term when the Document Type is a POS Sales Order.

So when we hit the Complete button to process the sale, it will no longer be a Standard Order, but converted into a Paid Invoice status, thus a C_Invoice (database level) record is created and its new Target Document Type is assumed which is an AR Invoice Indirect.

Since it is a Cash Sales, a Cash Journal entry is automatically created. Don't believe me? Then go to the Cash Journal window under Menu and voila! You find a new book created today and every sales transaction is created automatically until the day's close.

Also, if you go the Invoice (Customer) window, you will see corresponding records on the AR Invoice Indirect I talked about. Notice that they are already processed! That's no wonder because cash sales need not have an invoice, which may explain the term 'invoice indirect'. So therefore, the rule of every payment is only via invoices seems a consistent practice in the Compiere Application.

For End Of Day (EODA) operations, the cashier can print out her Invoice Detail from the Menu and count her cash to tally up for submission upstairs. We used the powerful inbuilt Report Generator to on-the-fly configure the report to show the important columns and provide the necessary sort order and summary totals.

The Accounts Dept. will then vet the report and close the Cash Journal. Later a transfer to bank account is done by creating another Cash Journal transfer entry.

Besides the Accounts Dept., the Operations team can access the Product Inventory to check on its stocks movements and act accordingly without delay as such information is real time and efficiently pulled out. This workflow can also be automatically done via Reorder Quantity setting.

Throughout this operational process, it becomes clear to us that the Client now has the tool to monitor its operations and control any risk of omitted acts. Full integration and well processed accounting entries also allow easy audit trail and track.

Final Touches

Breaking the Time Barrier

When I first demonstrated the POS System at the cafeteria counter, the Sports Centre owner was at hand to time me with a stopwatch, on how long he has to wait for the receipt to print. It took me at least a minute. The owner keeps shaking his head, muttering that he should go to a road-side hawker to experience less excruciation.

Thus I have to pull out all the stops. I really went into taking out fields and rearranging the screen from clutter and excess baggage. I reprogrammed APanel.java where the F7 and F9 function key acts as one touch keys to switch between upper and lower tabs. And there was Robin chipping in his two cents by getting the detail row out for added visual comfort to the cashiers.

But much of substantial time savings also came from outside Compiere's scope itself. It involves arranging your product IDs in an intuitive manner. Noticing that there are less than a hundred items that are been sold throughout the Sports Centre, I made a product ID rule that each product will carry a 2 digit unique ID. There must also not be any 3 digit or longer code that can dilute that numerical value rule.

Thus without needing a barcode scanner but a memorising of these codes placed in logical categories allows the staff to achieve efficiency that stop the clock within 5 secs before the receipt started printing. This was a self-admission by the Client that even amazes me.

I even designed a visual training cue to train those keyboard and mouse-challenged users. (See the next page).

We are especially thrilled by the Client's approval of the fact that Compiere's integrativeness brings about fingertip speed of pulling out real-time information and end of day reporting that group their performance according to cashier, counter and products sold.

Our premier client, No Pain No Gain Sports Centre's owner is also planning a much bigger facility covering 8 acres in CyberJaya soon, and may link up this two centres via broadband. We shouldn't worry then and will try out another aspect of Compiere which is the Replication capability and the WebStore.

- COMING SOON – FIXED ASSETS AND PAYROLL -

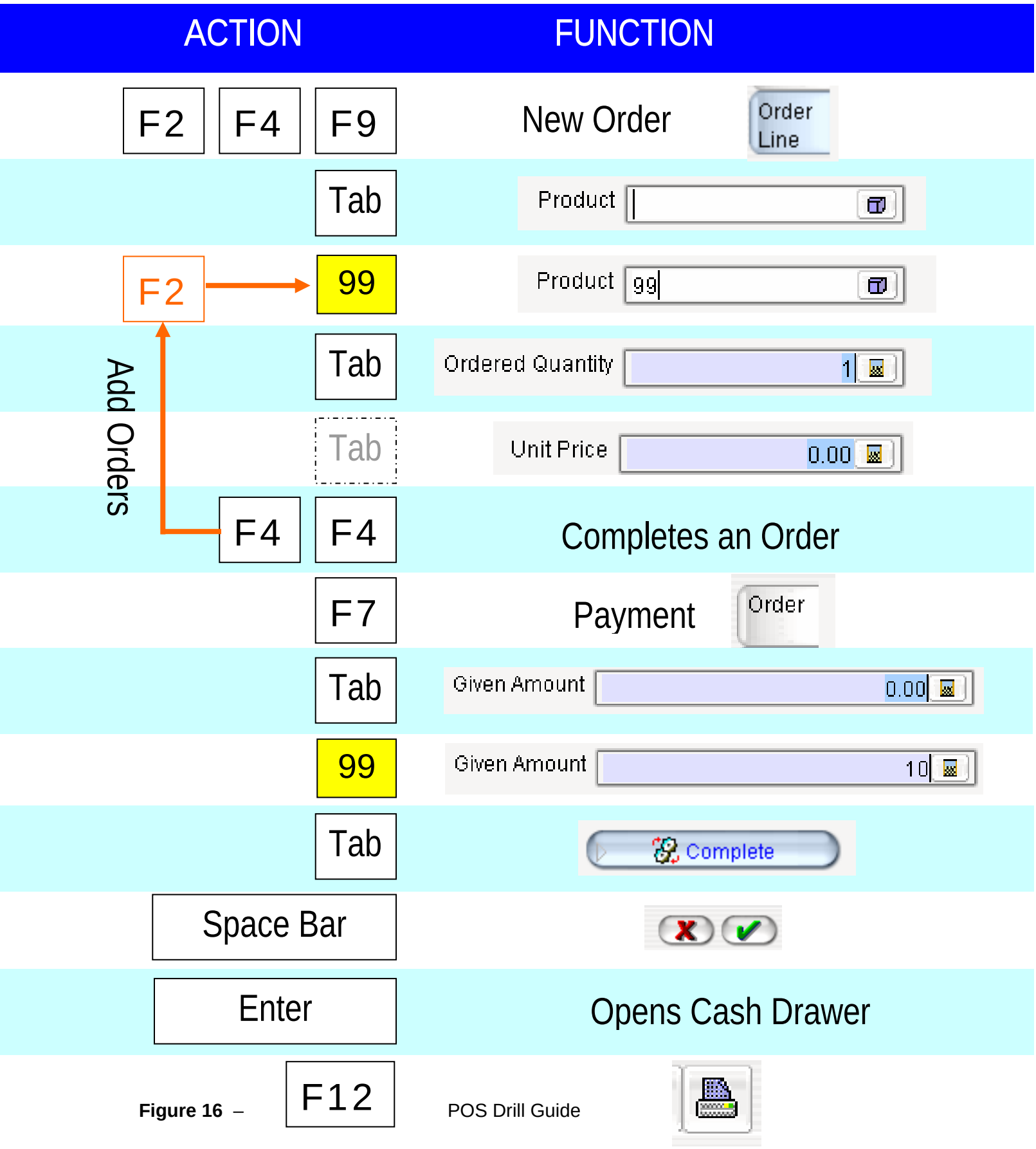

Figure 16 –