WORKFLOW

The Soul Of An ERP

Written in Cameroon

Central Africa

20th August

red1@red1.org

version 1.2

The 3 Laws of red1:

Information is Free

YOU HAVE TO KNOW

People are Not

YOU HAVE TO PAY

Contributors are Priceless

YOU HAVE TO BE

TABLE OF CONTENTS

With Simon Legah, President of Landmark University, Buea, and Dr. Stanley Mungwe, CEO of IT Kamer, Germany, and staff.
Stanley, who is also the Director of the Media and IT School under Landmark University, brought me to Germany and then Cameroon to build a support team for FOSS projects that can contribute not just to the iDempiere Free ERP project but trying to reverse the debilitating critical brain drain in Cameroon. I dedicate this free document to him and his people future that he is struggling for.
Photo taken in Yaounde, capital city of Cameroon where we had a series of conferences and TV and radio interviews.

WORKFLOW IN ERP

Defining Business Process

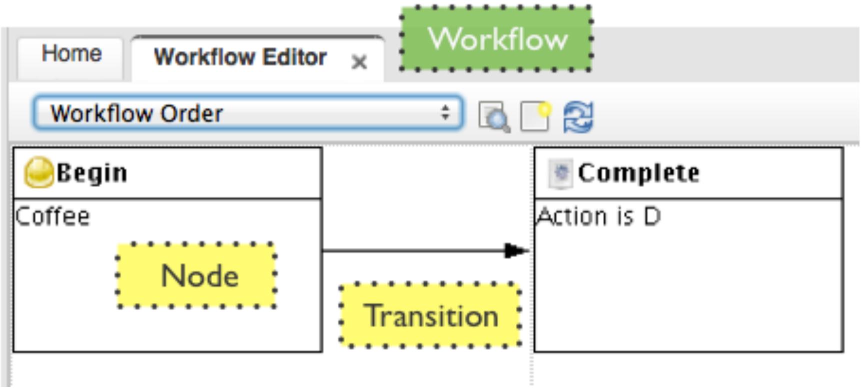
There are few ways in which the iDempiere Free ERP handles business logic:

1. *Document Event* happening in the Document Model

2. *Model Validator* intercepting such events

3. *Workflow* as part of a process within a Document Model

Of all three, *Workflow* is probably the most visible as it is accessible via the *Workflow Window* and the *Workflow Editor*. Whereas the others are more embedded within the model code and validator code themselves.



Above, is the present Workflow Editor showing Nodes (in boxes) linked by Transition (arrowed line). By pressing the Zoom icon, it will open up the standard Workflow window which also potrays the Condition tab.

In this work sponsored by SYSNOVA, I further improved the work of Compiere with the Wait (Schedule) step and the work of Low Heng Sin in the ZK UI Editor, with Node Action properties setting and Condition properties creation. My work is submitted for peer review via JIRA trackers, IDEMPIERE-2022 (Wait-Schedule) and IDEMPIERE-2080 (Editor improvement).

The World Out There

Having a Workflow engine is the craze with many ERP applications but if we look again closer it is not that simple as it seems. Take for instance the most advanced rendition out there given by JBoss Business Process Management or JBPM and the Drools Project, where normal English statements are made by lay users and then resolved into a logical rules set to be applied into the ERP. You can Google for 'Drools challenges / issues' and see the long path that still lies ahead before a true Artificial Intelligence arrives at the work-place. ERP is very complex is even more complex to integrate with such intelligence.

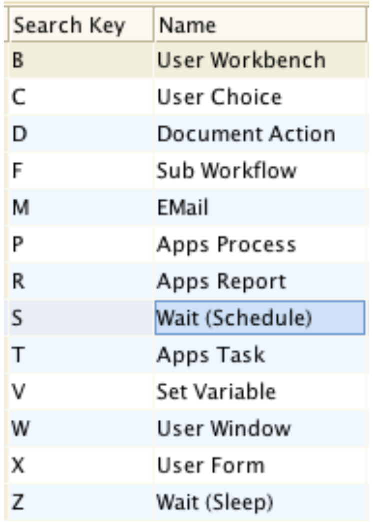
What iDempiere inherited from Compiere was an elementary 'from the ground up' Workflow engine that follows WfMc (Workflow Management Coalition) framework. Compiere already deals with every Document Model as DocStatus in ERP-speak with the events of *Start, Prepare, Complete and In-Progress, Reverse,* or *Void*. The WfMc framework adds a *Node-Transition-Condition* aspect to the document affair.

iDempiere is more sound with its freedom source nature easily exposed for common developers to debug, improve on to fit the most challenging of requirements from real world business apps. Low Heng Sin having worked a lot on the ZK Ajax user interface, has also refactored the Workflow Editor with important interfaces for which I could further explore a more comprehensive capability that allows complete creation of a whole Workflow model right past the Transition tab to the Condition tab without leaving the graphical editor.

IDEMPIERE-2022

The above JIRA tracker sets to improve the present Workflow Node' Action which are shown on the right. Note the highlighted Wait (Schedule) which is the extra property requested by this tracker.

When that action is selected, a column field appears to allow the selection of a date field, such as Date Promised or Date Ordered under the C_Order table (where the case might be for a workflow that acts on Sales Orders). This extension required the modification of the following model and core code.



Search Key	Name
B	User Workbench
C	User Choice
D	Document Action
F	Sub Workflow
M	EMail
P	Apps Process
R	Apps Report
S	Wait (Schedule)
T	Apps Task
V	Set Variable
W	User Window
X	User Form
Z	Wait (Sleep)

1. Table changed: AD_WF_Node extended with Wait (Schedule) field.

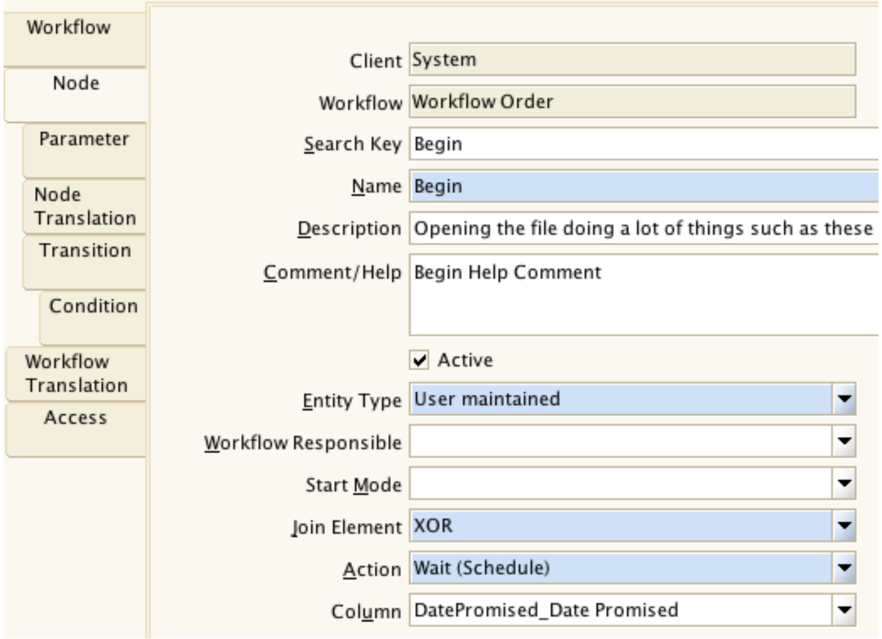
2. Code changed:

   a. MWFActivity.java

   b. WorkflowProcessor.java

For exact script and diff of the changes refer to IDEMPIERE-2022 online.

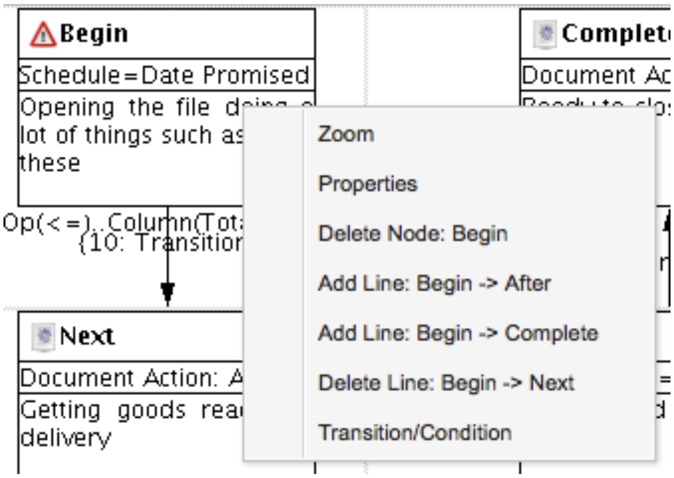
The extra Action option is seen here selected and its corresponding Column value of Date Promised selected. That means this node will suspend the processing of the associated document until the date specified is reached.



Client	System
Workflow	Workflow Order
Search Key	Begin
Name	Begin
Description	Opening the file doing a lot of things such as these
Comment/Help	Begin Help Comment
☑ Active	
Entity Type	User maintained
Workflow Responsible	
Start Mode	
Join Element	XOR
Action	Wait (Schedule)
Column	DatePromised_Date Promised

IDEMPIERE-2080

This tracker seek to improve the Workflow Editor to create on-the-fly, the Condition tab and its properties besides setting the Node Action and Column properties. It also sets the description field in the Transition tab. There is no metadata change but there is change to core code:



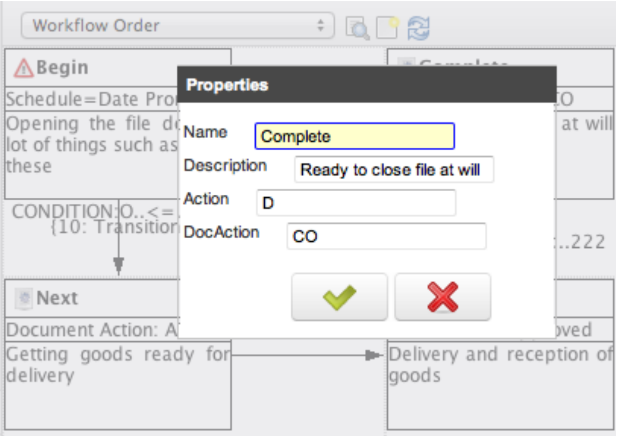
a. WFEditor.java

b. WFPopupItem.java

c. WFNodeWidget.java

d. WorkflowGraphScene.java

e. MWFNode.java


Workflow Order
Begin
Schedule=Date Pro
Opening the file do lot of things such as these
CONDITION:O..<=
{10: Transition
Next
Document Action: A
Getting goods ready for delivery
Delivery and reception of goods
Properties

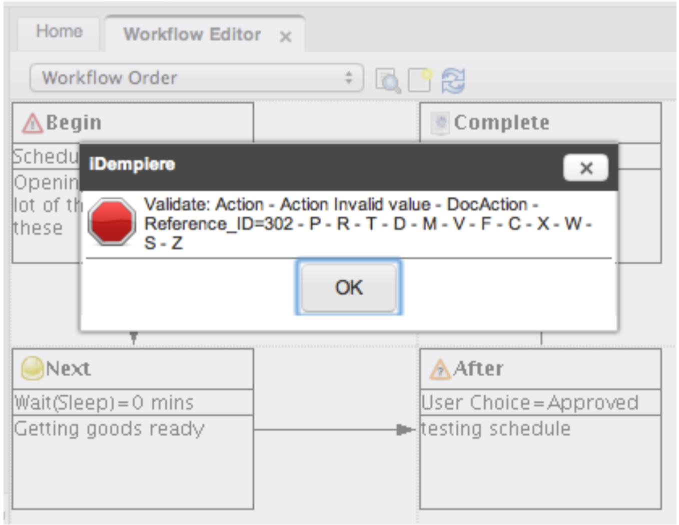
Name: Complete

Description: Ready to close file at will

Action: D

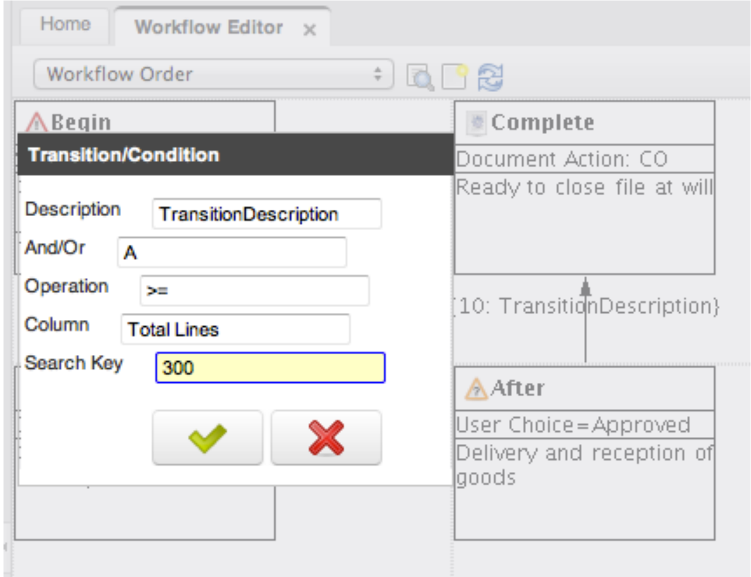
DocAction: CO

After applying the code change, any node with a transition arrow leading from it to another node, when clicked upon will show an additional feature, 'Transition/ Condition'. When selecting the Properties feature, there are two new fields that can be set in the pop-up box as shown on the right, namely 'Action' and depending on the value in the Action field, will then display accordingly the next field. Values set in these two new fields have to be native, such as shown in the example. If a wrong value is given, an advice will appear.



Validate: Action - Action Invalid value - DocAction - Reference_ID=302 - P - R - T - D - M - V - F - C - X - W - S - Z

Selecting the Transition/Condition feature, will display the following pop-up box.

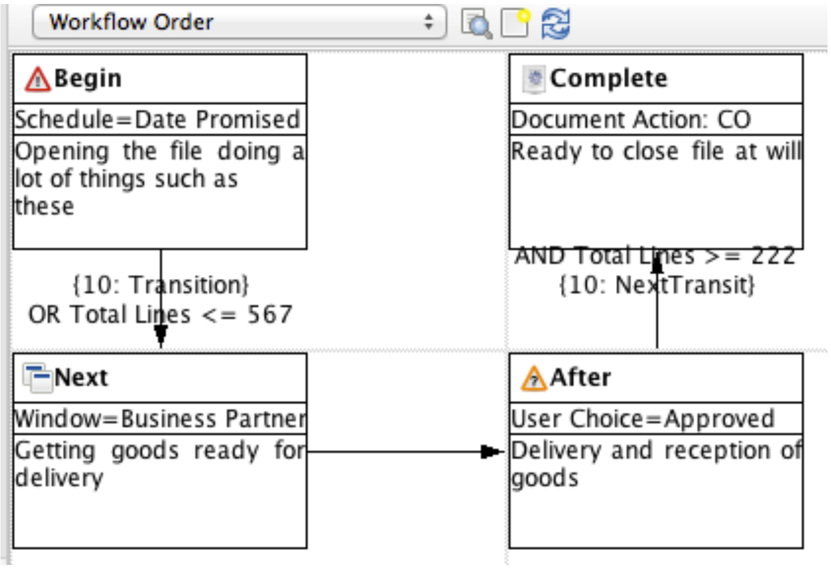


Transition/Condition

Description: TransitionDescription
And/Or: A
Operation: >=
Column: Total Lines
Search Key: 300

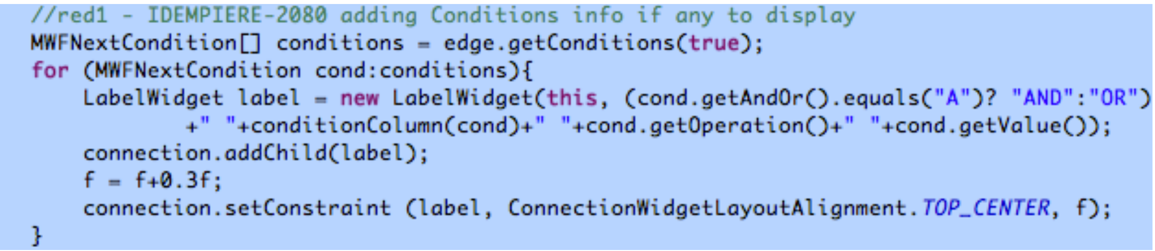
✓ ✗

User can now enter the Description of the Transition tab, then the Condition tab values such as And/Or, Operation and Value (Search Key). If the Condition tab is blank, it shall create automatically. Thus the user mostly need not zoom into the Workflow window to fill in such values. Upon completion, the editor will display the new information as shown below.



This maximizes the visual user-friendliness of the Workflow Editor and minimizes the need to go into the Workflow window except for more finer setting.

The display of `OR Total Lines <= 567` on the first Transition line is done by this code in WorfklowGraphScene.java at the attachEdgeWidget method:



```
//red1 - IDEMPIERE-2080 adding Conditions info if any to display
MWFNextCondition[] conditions = edge.getConditions(true);
for (MWFNextCondition cond:conditions){
    LabelWidget label = new LabelWidget(this, (cond.getAndOr().equals("A")? "AND":"OR")
            +" "+conditionColumn(cond)+" "+cond.getOperation()+" "+cond.getValue());
    connection.addChild(label);
    f = f+0.3f;
    connection.setConstraint (label, ConnectionWidgetLayoutAlignment.TOP_CENTER, f);
}
```

According to a World Bank report from 2012, more than 70% of Cameroon's graduates are underemployed, another way of saying they find no jobs that fit what they studied for. Many either do menial jobs or run overseas where able. However, they are highly talented, eloquent in English and French, with many possessing a string of college qualifications even Masters Degrees if not Doctorates. The country lacked many modern trappings, hampered further by poor transportation, proper water and power supply, what more Internet access which has only 6.4% penetration in a population of around 23 million. I struggled with the intermittent 2G mobile facility to continue working online from here.

USING WORKFLOW IN IDEMPIERE

Lessons From The Past

You may pick up previous published tutorials on how to workout an approval process:

http://red1.org/adempiere/viewtopic.php?f=45&t=1801 from Jorg Janke, Compiere itself.

You can also look at an old wiki set of
http://www.adempiere.com/How_to_Activate_Document_Approval_Workflow and

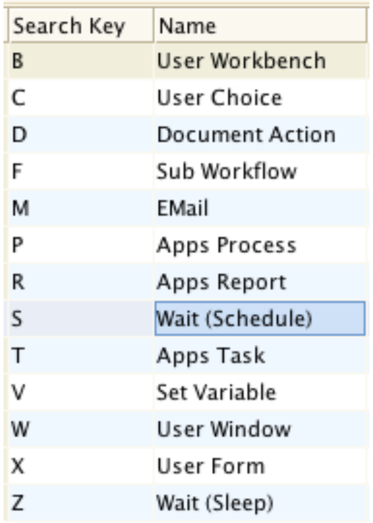
http://www.adempiere.com/How_to_Configure_Dynamic_Approval_Workflow.

Work In Progress

This is still a work in progress for peer review and further enhancements are definitely possible. However, as the changes are to the core, which may carry impact to any other use of the affected model and classes, the work done thus far is for a POC (proof of concept) in order that other ideas from the community can be iterated firstly.

The next steps is to give the field input a better combo pull down list instead of a direct memory input. For now, the Node Action reference list guide here can be of help. Use the one character Search Key for Node Properties Action setting. Presently defined Workflows in iDempiere seems to be just requiring W, Z, D, C and the new S.

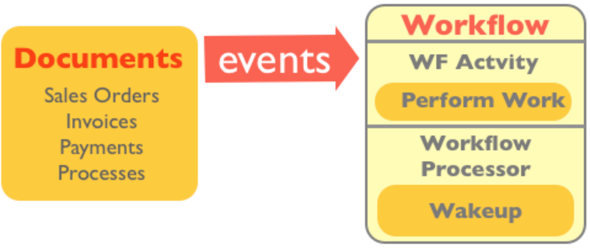
Nevertheless, I have added code for it to handle P, R, V, X and F! So do enjoy exploiting these other unused options. If you have ideas to improve further just study the code and drop me an email giving full reference of real world use that I might be interested to extend further the action processing part.



Search Key	Name
B	User Workbench
C	User Choice
D	Document Action
F	Sub Workflow
M	EMail
P	Apps Process
R	Apps Report
S	Wait (Schedule)
T	Apps Task
V	Set Variable
W	User Window
X	User Form
Z	Wait (Sleep)

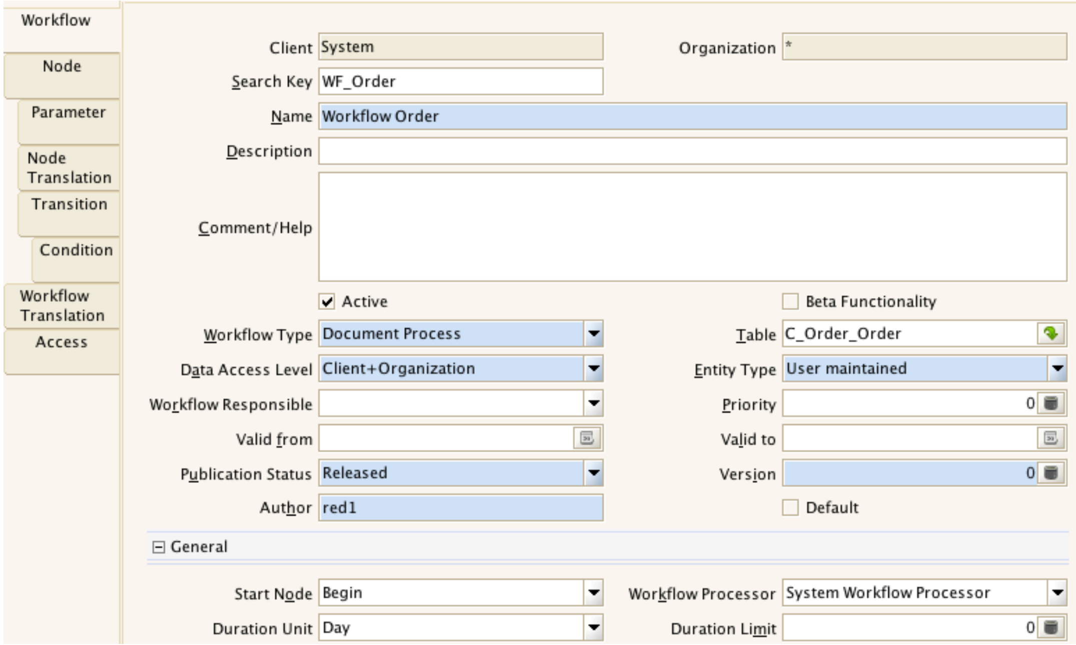
If you get a null pointer exception during wrong entry of the subsequent resulting column, you may need to do a Cache Reset before continuing.

How Workflow Works



Any process in a document can activate a Workflow which has two main actors - the Workflow Activity and Workflow Processor.

Firstly, many events can spawn a workflow process. Usually in an ERP, it is the important documents like Sales Orders that have events for example the process button that is associated with a workflow. In my test case, I created a new Workflow that intercepted the C_Order table used by both Sales Orders and Purchase Orders, as shown below:



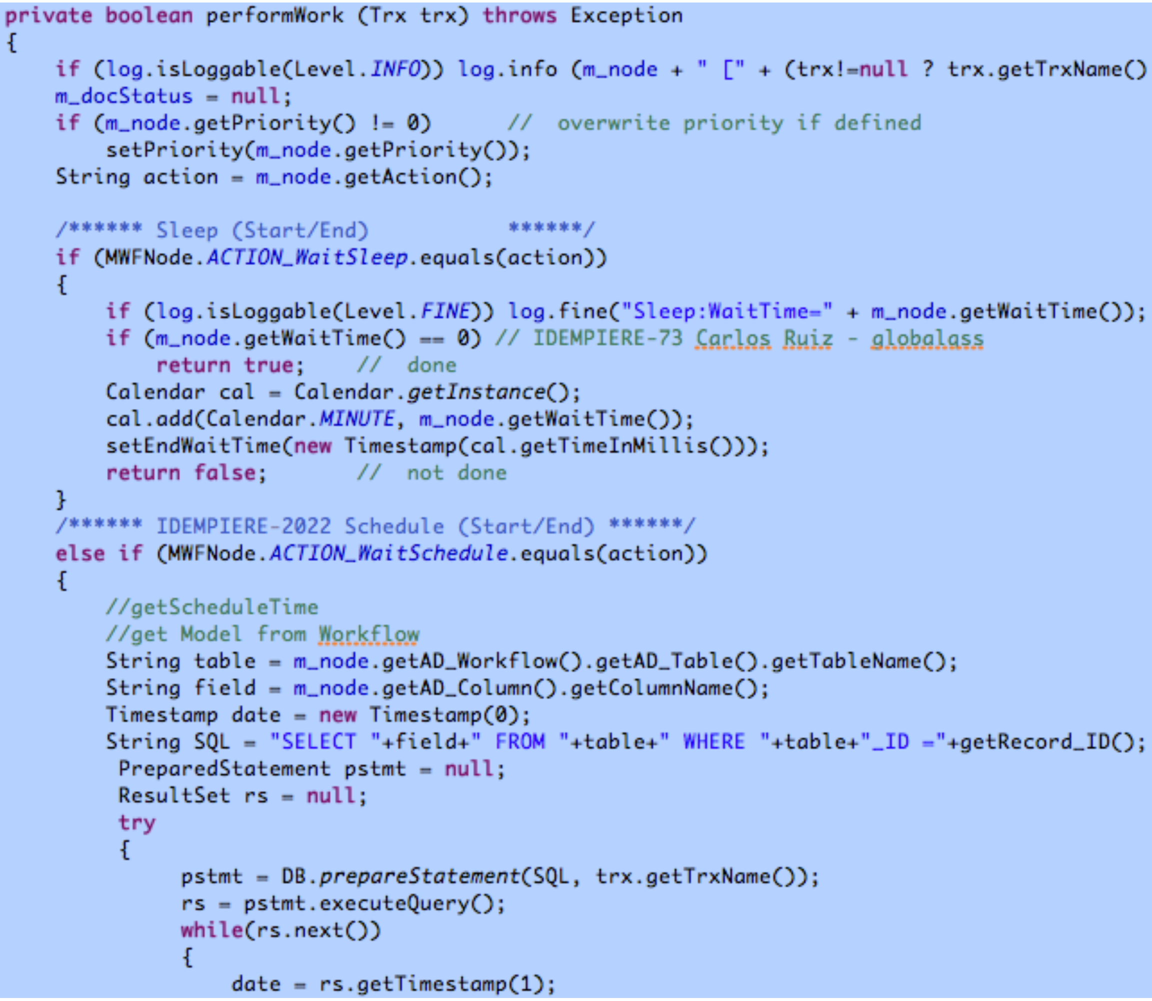
Client	System	Organization	*
Search Key	WF_Order		
Name	Workflow Order		
Description			
Comment/Help			
	☑ Active		□ Beta Functionality
Workflow Type	Document Process	Table	C_Order_Order
Data Access Level	Client+Organization	Entity Type	User maintained
Workflow Responsible		Priority	0
Valid from		Valid to	
Publication Status	Released	Version	0
Author	red1		□ Default

⊟ General

Start Node	Begin	Workflow Processor	System Workflow Processor
Duration Unit	Day	Duration Limit	0

It is also to be handled by the Workflow Processor. More on how the processor work later.

After a workflow activity is created, it will *Perform Work* to take on each node. It will analyse first the two Wait actions i.e. the Sleep and Schedule options where a Date timestamp will be calculated and set as the EndWaitTime.

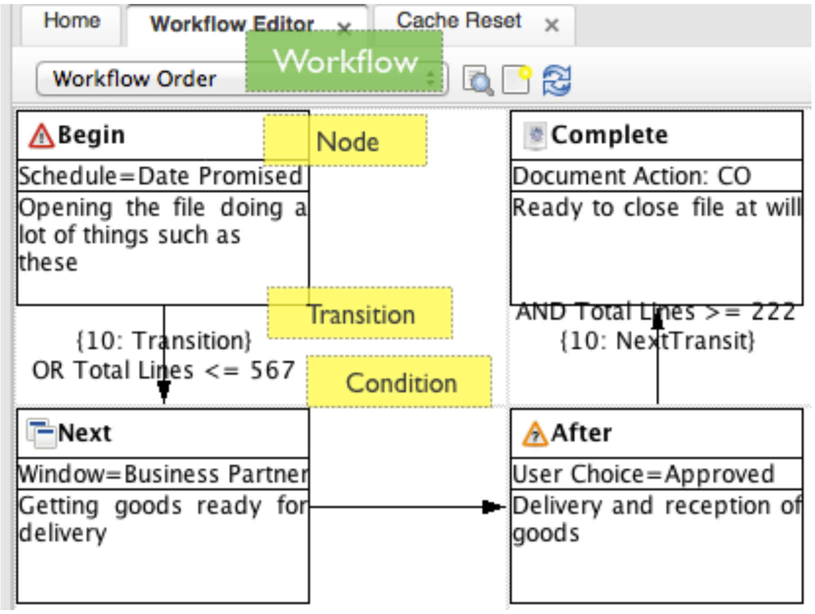


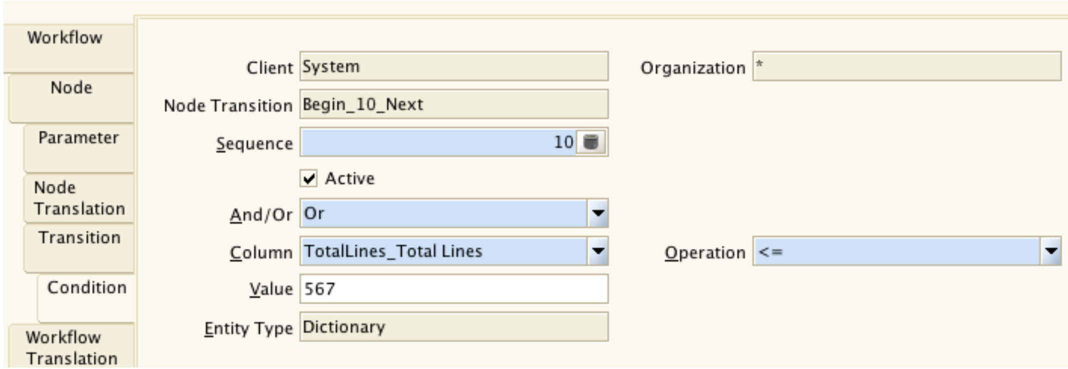
```
private boolean performWork (Trx trx) throws Exception
{
    if (log.isLoggable(Level.INFO)) log.info (m_node + " [" + (trx!=null ? trx.getTrxName()
    m_docStatus = null;
    if (m_node.getPriority() != 0)      //  overwrite priority if defined
        setPriority(m_node.getPriority());
    String action = m_node.getAction();

    /****** Sleep (Start/End)           ******/
    if (MWFNode.ACTION_WaitSleep.equals(action))
    {
        if (log.isLoggable(Level.FINE)) log.fine("Sleep:WaitTime=" + m_node.getWaitTime());
        if (m_node.getWaitTime() == 0) // IDEMPIERE-73 Carlos Ruiz - globalqss
            return true;    //  done
        Calendar cal = Calendar.getInstance();
        cal.add(Calendar.MINUTE, m_node.getWaitTime());
        setEndWaitTime(new Timestamp(cal.getTimeInMillis()));
        return false;       //  not done
    }
    /****** IDEMPIERE-2022 Schedule (Start/End) ******/
    else if (MWFNode.ACTION_WaitSchedule.equals(action))
    {
        //getScheduleTime
        //get Model from Workflow
        String table = m_node.getAD_Workflow().getAD_Table().getTableName();
        String field = m_node.getAD_Column().getColumnName();
        Timestamp date = new Timestamp(0);
        String SQL = "SELECT "+field+" FROM "+table+" WHERE "+table+"_ID ="+getRecord_ID();
         PreparedStatement pstmt = null;
         ResultSet rs = null;
         try
         {
              pstmt = DB.prepareStatement(SQL, trx.getTrxName());
              rs = pstmt.executeQuery();
              while(rs.next())
              {
                  date = rs.getTimestamp(1);
```

Each node action corresponds to the Workflow model of Node, Transition, Condition. On the next page we have that relationship laid out to maintain a high overview of things as possible.

With the new enhancement, we can see each node action such as Schedule=Date Promised or User Choice=Approved in the node image. The transition line is now added with condition properties such as OR Total Lines <= 567. The actual condition tab is displayed below:



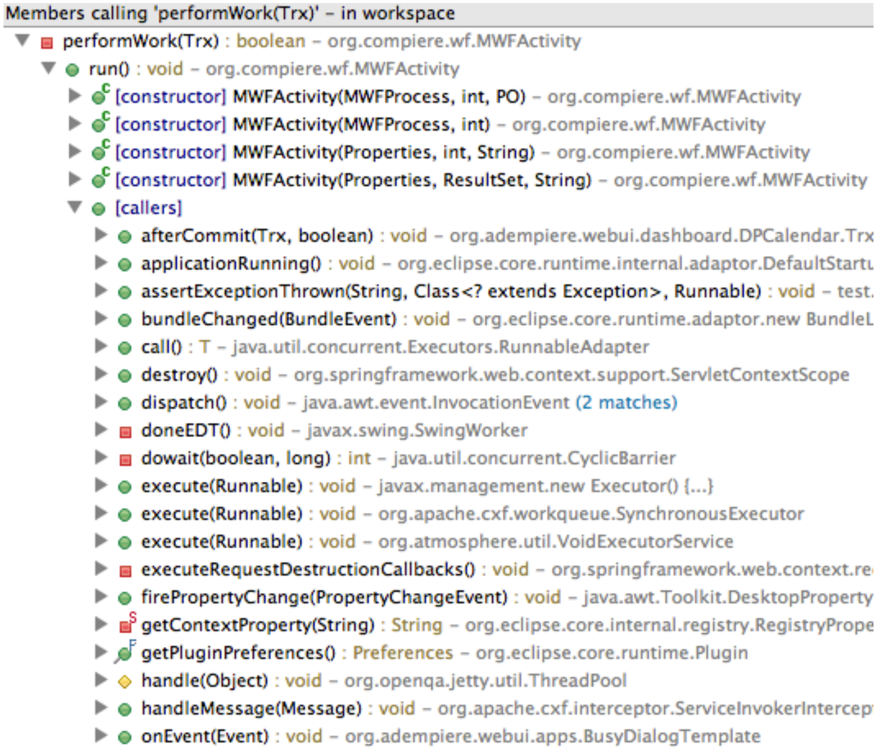


Client	System	Organization	*
Node Transition	Begin_10_Next		
Sequence	10		
	☑ Active		
And/Or	Or		
Column	TotalLines_Total Lines	Operation	<=
Value	567		
Entity Type	Dictionary		

We can change the node and condition properties on the fly in the graphical editor but only from the ZK Ajax UI as the enhancement is done there. That is illustrated on page 7 and 8. More on the code behind condition later. The display of a single sentence of the condition is highly readable and the ability to just change it on the fly should be the ultimate experience thus far on this free ERP.

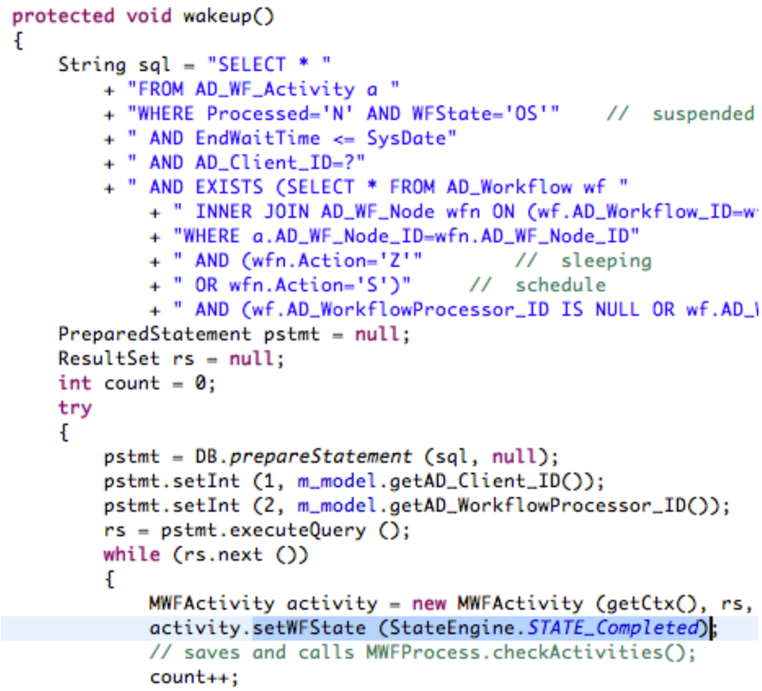
Workflow Processor

As shown, the new *Wait(Schedule)* action node is now incorporated to set the *EndWaitTime* according to the date defined. This can be studied in the MWFActivity.java class. (It is called by many members in the source code):



```
Members calling 'performWork(Trx)' – in workspace
▼ performWork(Trx) : boolean – org.compiere.wf.MWFActivity
  ▼ run() : void – org.compiere.wf.MWFActivity
    ▶ [constructor] MWFActivity(MWFProcess, int, PO) – org.compiere.wf.MWFActivity
    ▶ [constructor] MWFActivity(MWFProcess, int) – org.compiere.wf.MWFActivity
    ▶ [constructor] MWFActivity(Properties, int, String) – org.compiere.wf.MWFActivity
    ▶ [constructor] MWFActivity(Properties, ResultSet, String) – org.compiere.wf.MWFActivity
    ▼ [callers]
      ▶ afterCommit(Trx, boolean) : void – org.adempiere.webui.dashboard.DPCalendar.Trx
      ▶ applicationRunning() : void – org.eclipse.core.runtime.internal.adaptor.DefaultStartu
      ▶ assertExceptionThrown(String, Class<? extends Exception>, Runnable) : void – test.
      ▶ bundleChanged(BundleEvent) : void – org.eclipse.core.runtime.adaptor.new BundleL
      ▶ call() : T – java.util.concurrent.Executors.RunnableAdapter
      ▶ destroy() : void – org.springframework.web.context.support.ServletContextScope
      ▶ dispatch() : void – java.awt.event.InvocationEvent (2 matches)
      ▶ doneEDT() : void – javax.swing.SwingWorker
      ▶ dowait(boolean, long) : int – java.util.concurrent.CyclicBarrier
      ▶ execute(Runnable) : void – javax.management.new Executor() {...}
      ▶ execute(Runnable) : void – org.apache.cxf.workqueue.SynchronousExecutor
      ▶ execute(Runnable) : void – org.atmosphere.util.VoidExecutorService
      ▶ executeRequestDestructionCallbacks() : void – org.springframework.web.context.re
      ▶ firePropertyChange(PropertyChangeEvent) : void – java.awt.Toolkit.DesktopProperty
      ▶ getContextProperty(String) : String – org.eclipse.core.internal.registry.RegistryPrope
      ▶ getPluginPreferences() : Preferences – org.eclipse.core.runtime.Plugin
      ▶ handle(Object) : void – org.openqa.jetty.util.ThreadPool
      ▶ handleMessage(Message) : void – org.apache.cxf.interceptor.ServiceInvokerIntercep
      ▶ onEvent(Event) : void – org.adempiere.webui.apps.BusyDialogTemplate
```

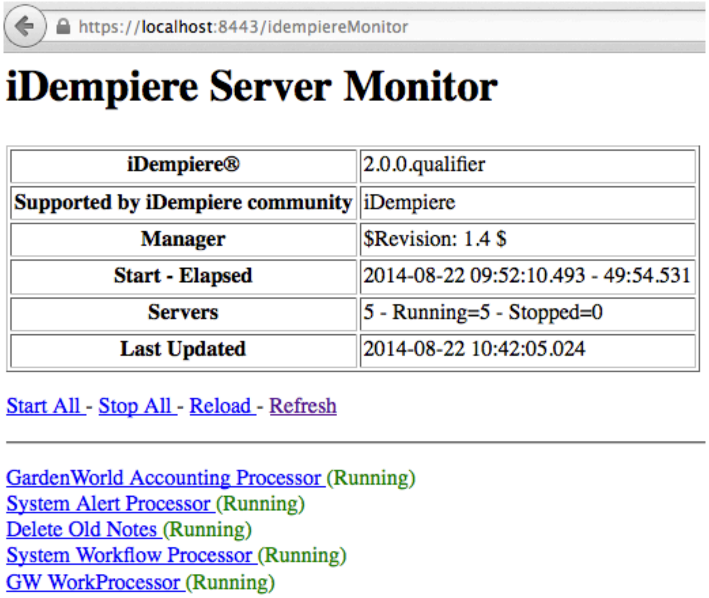
Before it can perform work, the two Wait actions can put it on hold until the *EndWaitTime* has come to pass. How is this date-time point checked? It is done by another class, WorkflowProcessor.java during *WakeUp*: This processor is controlled by the server as shown on next page.



```
protected void wakeup()
{
    String sql = "SELECT * "
        + "FROM AD_WF_Activity a "
        + "WHERE Processed='N' AND WFState='OS'"    //  suspended
        + " AND EndWaitTime <= SysDate"
        + " AND AD_Client_ID=?"
        + " AND EXISTS (SELECT * FROM AD_Workflow wf "
            + " INNER JOIN AD_WF_Node wfn ON (wf.AD_Workflow_ID=w
            + "WHERE a.AD_WF_Node_ID=wfn.AD_WF_Node_ID"
            + " AND (wfn.Action='Z'"        //  sleeping
            + " OR wfn.Action='S')"     //  schedule
            + " AND (wf.AD_WorkflowProcessor_ID IS NULL OR wf.AD_
    PreparedStatement pstmt = null;
    ResultSet rs = null;
    int count = 0;
    try
    {
        pstmt = DB.prepareStatement (sql, null);
        pstmt.setInt (1, m_model.getAD_Client_ID());
        pstmt.setInt (2, m_model.getAD_WorkflowProcessor_ID());
        rs = pstmt.executeQuery ();
        while (rs.next ())
        {
            MWFActivity activity = new MWFActivity (getCtx(), rs,
            activity.setWFState (StateEngine.STATE_Completed);
            // saves and calls MWFProcess.checkActivities();
            count++;
```

The 'S' (WaitSchedule) action setting is added to the check if the system time of the computer or server has passed the date timestamp of EndWaitTime. If so, then the activity table, AD_WF_Activity will change its state to completed.

Here we are using the iDempiere Server Monitor called by pressing the icon on the top left of the http://localhost:8080 opening page display when we run iDempiere Server. You can see the System Workflow Processor with its status of Running. You can press on its link to get to it and check its running time and during testing you can make it run on demand to see if your workflow case works.



https://localhost:8443/idempiereMonitor

iDempiere Server Monitor

iDempiere®	2.0.0.qualifier
Supported by iDempiere community	iDempiere
Manager	$Revision: 1.4 $
Start - Elapsed	2014-08-22 09:52:10.493 - 49:54.531
Servers	5 - Running=5 - Stopped=0
Last Updated	2014-08-22 10:42:05.024

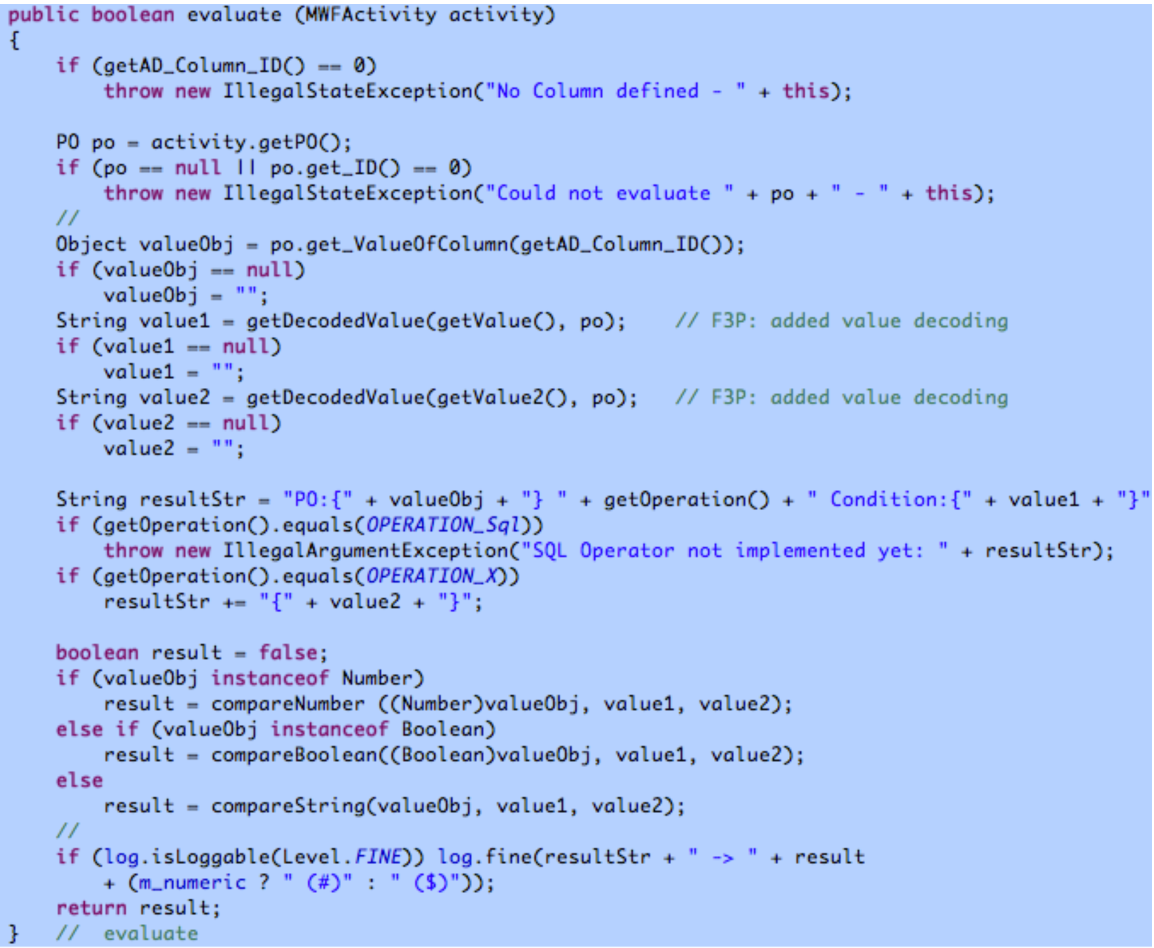
Start All - Stop All - Reload - Refresh

GardenWorld Accounting Processor (Running)
System Alert Processor (Running)
Delete Old Notes (Running)
System Workflow Processor (Running)
GW WorkProcessor (Running)

About half the population lives under the poverty line (USD1.25 per day per person). 70% of Cameroon lives off the farms under challenging conditions. Begging in the streets is extremely rare and most take to the streets as petty traders. Here a woman sells bottles of nuts with her two boys in attendance.

Evaluate Condition

As mentioned, there is the last tab feature in the Workflow which is a killer that is the *Condition* which is checked also before executing any node action. The next code snapshot shows how the condition evaluation is done. It is called eventually during wake up.



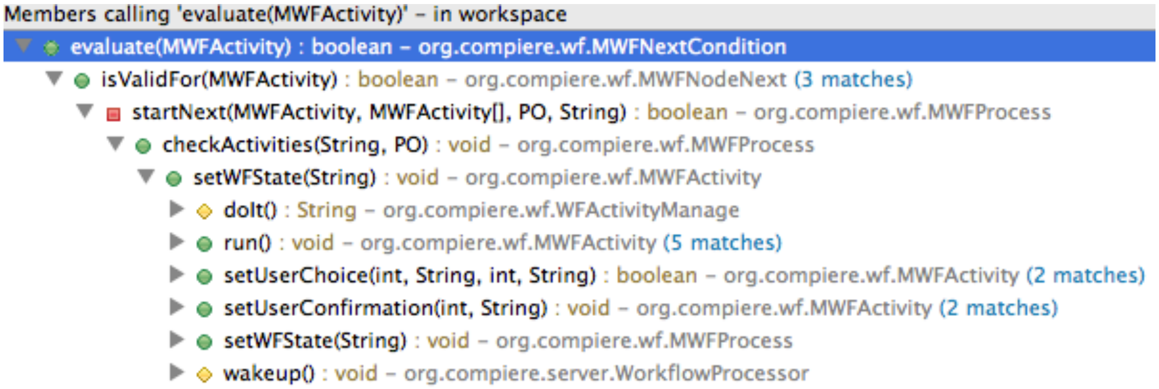
```java
public boolean evaluate (MWFActivity activity)
{
    if (getAD_Column_ID() == 0)
        throw new IllegalStateException("No Column defined - " + this);

    PO po = activity.getPO();
    if (po == null || po.get_ID() == 0)
        throw new IllegalStateException("Could not evaluate " + po + " - " + this);
    //
    Object valueObj = po.get_ValueOfColumn(getAD_Column_ID());
    if (valueObj == null)
        valueObj = "";
    String value1 = getDecodedValue(getValue(), po);    // F3P: added value decoding
    if (value1 == null)
        value1 = "";
    String value2 = getDecodedValue(getValue2(), po);   // F3P: added value decoding
    if (value2 == null)
        value2 = "";

    String resultStr = "PO:{" + valueObj + "} " + getOperation() + " Condition:{" + value1 + "}"
    if (getOperation().equals(OPERATION_Sql))
        throw new IllegalArgumentException("SQL Operator not implemented yet: " + resultStr);
    if (getOperation().equals(OPERATION_X))
        resultStr += "{" + value2 + "}";

    boolean result = false;
    if (valueObj instanceof Number)
        result = compareNumber ((Number)valueObj, value1, value2);
    else if (valueObj instanceof Boolean)
        result = compareBoolean((Boolean)valueObj, value1, value2);
    else
        result = compareString(valueObj, value1, value2);
    //
    if (log.isLoggable(Level.FINE)) log.fine(resultStr + " -> " + result
        + (m_numeric ? " (#)" : " ($)"));
    return result;
}   //  evaluate
```

The way the wake up reaches here is quite elaborate:



```
Members calling 'evaluate(MWFActivity)' - in workspace
▼ evaluate(MWFActivity) : boolean - org.compiere.wf.MWFNextCondition
    ▼ isValidFor(MWFActivity) : boolean - org.compiere.wf.MWFNodeNext (3 matches)
        ▼ startNext(MWFActivity, MWFActivity[], PO, String) : boolean - org.compiere.wf.MWFProcess
            ▼ checkActivities(String, PO) : void - org.compiere.wf.MWFProcess
                ▼ setWFState(String) : void - org.compiere.wf.MWFActivity
                    ▶ doIt() : String - org.compiere.wf.WFActivityManage
                    ▶ run() : void - org.compiere.wf.MWFActivity (5 matches)
                    ▶ setUserChoice(int, String, int, String) : boolean - org.compiere.wf.MWFActivity (2 matches)
                    ▶ setUserConfirmation(int, String) : void - org.compiere.wf.MWFActivity (2 matches)
                    ▶ setWFState(String) : void - org.compiere.wf.MWFProcess
                    ▶ wakeup() : void - org.compiere.server.WorkflowProcessor
```

As shown in the node action chart at the top of page 6, this process can govern any business logic, presumably as well as any event validator. Any logic is a matter of flow, condition and action. From here, I hope you will make use of the true power of this workflow more often.

There are more than 250 tribes with its own language and traditional customs. Here at a funeral one tribal group is presenting their peculiar costume and dance. They are mostly peaceful and Cameroon has never had a civil conflict since its independence in 1960. Cameroon is known as Africa in miniature due to its geographical makeup. Though it is still a poor country, it is relatively better off than those in the rest of Central Africa. The government has embarked on an emerging Cameroon vision by the year 2035.