Using Tools to Represent Appraisal Status Flow (HR module "Objectives Settings and Appraisals") as Diagram

Applies to:

SAP ECC 6.0, Enterprise Extension 2.0

Summary

You define the status flow for appraisal documents according to the different phases in the appraisal process. After this operation, you get a set of steps and you can go from one step to another if different conditions are fulfilled. From a mathematical point of view, the status flow is a graph and the steps are the nodes of the graph. It's a bit difficult to see the big picture of your flow when the steps and the transitions are stored and displayed as a table. What I propose here is to extract the status flow and export it into a tool that allows you to see the whole process in a visual way.

Author: Pierre Godart

Company: Quintisys (http://www.quintisys.com)

Created on: 01 January 2009

Author Bio

Pierre Godart has over 8 year's experience developing solutions for SAP HR systems, both in the 'back end' R/3 HR environment and in ESS / MSS web applications. Pierre began his career with as lead technical consultants and developed SAP HR charting and reporting tools, as well as consulting on SAP HR projects.

Table of Contents

The appraisal status flow

Introduction

Each step in the status flow is identified by a status (and a sub-status). To pass from one step in the process to another, you need to do an action like “submit to first reviewer”. This action is the materialized in the form of a button and the right to execute the action is represented by the fact that the button is enabled or exists. This right is technically implemented via a (workflow) rule. So to resume, a step is made of:

- A status identifier (+ text)
- A sub-status identifier (+ text)

Via a push button and a rule, you get access to the next step; this transition can raise an event that in its turn will raise a workflow. So, a transition from one step to another step is characterized by:

- Push button
- Authorization/Access rule
- Workflow event

Table and data model

The setup of the appraisal status flow is performed via the transaction “PHAP_CATALOG”. You configure the process flow of the appraisal via the tab “Status flow” on the “Appraisal Template” (object VA). You can have a resume of all the steps and how to move from one to another via the button “Formatted report”

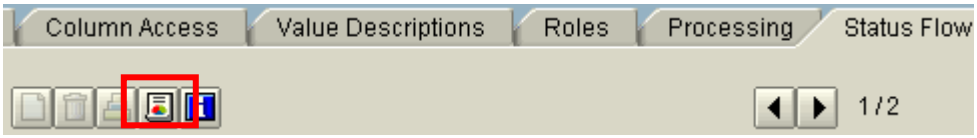


Figure 1 - Formatted report button

This gives you the big picture of the appraisal process flow in a tabular format:

Status Flow: Overview

Outbound Status	Outbound Substatus	Targ.Stat.	Target Substatus	Pushbutton Text	Authorized	Next Status
Simulated Status		In Process			All Users	Saved and Change to Display Mod
In Preparation		In Planning		Define Objectives	All Users	
In Preparation		In Planning		End Preparation	All Users	Saved and Exit Appraisal Documer
In Planning		In Review		Review	All Users	
In Planning		In Process	Part Appraisal in Process	Start Appraisal Process	All Users	Saved and Exit Appraisal Documer
In Planning		In Process		Execute	All Users	
In Review		In Process		Execute	All Users	
In Review		In Process	Part Appraisal in Process	Start Appraisal Process	All Users	Saved and Exit Appraisal Documer
In Review		In Review		Next Review	All Users	
In Process		Completed		Complete	All Users	Saved and Change to Display Mod
In Process	Part Appraisal in Process	In Process	Part Appraisal in Process	Close Part Appraisal	All Users	Saved and Change to Display Mod
Completed					All Users	

Figure 2 - Appraisal flow in tabular format

The above example is the standard template proposed for "Performance Feedback"

Now, technically, where can I get all this information? The main information needed can be found in infotype HRP5026 (and its extension HRT5026). Here is the corresponding structure:

Structure PT5026 Active

Short Description Appraisal Element Processing

Attributes | Components | Entry help/check | Currency/quantity fields

Predefined Type 1 / 11

Component	Component type	Short Description
OLD_STATUS	HAP_SOURCE_STATUS	Outbound Status
OLD_STATUS_SUB	HAP_SOURCE_STATUS_SUB	Outbound Substatus
NO_CHANGE	HAP_AP_NO_CHANGE	Flag: Changes in Appraisal Document Not Allowed
WF_RULE_ID	HAP_WF_RULE_ID	Person Authorized
BUTTON_ID	HAP_BUTTON_ID	Pushbutton
INTENSIFIED	HAP_BUTTON_INTENSIFIED	Pushbutton Highlighted
NEW_STATUS	HAP_TARGET_STATUS	Target Status
NEW_STATUS_SUB	HAP_TARGET_STATUS_SUB	Target Substatus
DOCUMENT_LEAVE	HAP_DOCUMENT_LEAVE	Next Status of Appraisal Document
WF_EVENT_ID	HAP_WF_EVENT_ID	Workflow Event
STATUS_NOTE	HAP_STATUS_NOTE_AVAILABILITY	Create Note

Figure 3 - Infotype 5026

Remark: the function module HRHAP_C_IT5026_READ should allow you to get easily access to the data (I saw it too late to test it, I used an OPENSQL command instead (bad!)).

Rendering tools for graph

Introduction

Here are the steps to get a beautiful graphic representing the appraisal status flow:

- Extract appraisal status flow data
- Convert these data to an output format corresponding to the chosen tool
- Download the converted data into a file
- Launch/open the graph rendering tool with the data file

I've used two great tools to render graphs (free download!): yEd and GraphViz. Let's have a little description of the tools.

yEd

yEd is a very powerful graph editor that can be used to quickly and effectively generate drawings and to apply automatic layouts to a range of different diagrams and networks. yEd is available as a free download with unrestricted functionality! I will describe more in details the way using this tool.

yEd Web Site

GraphViz

Graph visualization is a way of representing structural information as diagrams of abstract graphs and networks. Graphviz is open source graph visualization software. It has several main graph layout programs. The Graphviz layout programs take descriptions of graphs in a simple text language, and make diagrams in several useful formats such as images and SVG for web pages, Postscript for inclusion in PDF or other documents; or display in an interactive graph browser. (Graphviz also supports GXL, an XML dialect.)

GraphViz Web Site

The DOT file format

This file format is used by GraphViz. It is a quite simple text file format and the graph (flow) is described using the DOT language. More information can be found here.

The GML file format

You can use this file format when you work with yEd. It's also a quite simple text file format. Let's go deeper in his presentation.

Structure:

The main envelope is like this:

```
graph [
  comment ""
  directed 1
  IsPlanar 1

  -- HERE, you will find a list of node + their description --

  -- HERE, you will find a list of edge + their description --

]
```

The format of a node is like this:

```
  node [  id 19   label " Completed [ 5 , B  ] "  ]
```

Information: the identifier and the label of the node.

The format of an edge is like this:

```
    edge [  source  2   target 18
         label "       Button  Txt : Submit for approval
                     Button  Id  : ZSUBMIT
                     Rule    Id  : 99000011
                     Rule    Txt : Get the N+1, the N+2 and the backup
                     Event   Txt : STEP_XX
                     Event   Id  : EVENT4STEP_XX  "
       ]
```

Information: the source and destination of the link and its label.

Here is the GML file corresponding to the figure 2 (status flow overview of the template “Performance feedback”):

```
graph [
 comment "Template: Performance Feedback"
 directed 1
 IsPlanar 1
 node [  id 1   label " In Preparation [ 1 ,  ]                          "  ]
 node [  id 2   label " In Planning [ 2 ,  ]                             "  ]
 node [  id 3   label " In Review [ 3 ,  ]                               "  ]
 node [  id 4   label " In Process Part Appraisal in Process [ 4 , 1 ]   "  ]
 node [  id 5   label " In Process [ 4 ,  ]                              "  ]
 node [  id 6   label " Simulated Status [ 0 ,  ]                        "  ]
 node [  id 7   label " Completed [ 5 ,  ]                               "  ]
 edge [  source 1   target 2   label "   Button Txt: Define Objectives          Button  Id: OBJECTIVE" ]
 edge [  source 1   target 2   label "   Button Txt: Define Objectives          Button  Id: OBJECTIVE" ]
 edge [  source 2   target 3   label "   Button Txt: Review                     Button  Id: REVIEW    " ]
 edge [  source 2   target 4   label "   Button Txt: Start Appraisal Process    Button  Id: START     " ]
 edge [  source 2   target 5   label "   Button Txt: Execute                    Button  Id: PERFORM   " ]
 edge [  source 3   target 3   label "   Button Txt: Next Review                Button  Id: N_REVIEW  " ]
 edge [  source 3   target 4   label "   Button Txt: Start Appraisal Process    Button  Id: START     " ]
 edge [  source 3   target 5   label "   Button Txt: Execute                    Button  Id: PERFORM   " ]
 edge [  source 4   target 4   label "   Button Txt: Close Part Appraisal       Button  Id: COMPL_P   " ]
 edge [  source 5   target 7   label "   Button Txt: Complete                   Button  Id: COMPLETE  " ]
 edge [  source 6   target 5   label "   Button Txt:                            Button  Id:           " ]
 edge [  source 7   target 6   label "   Button Txt:                            Button  Id:           " ]
]
```

The ABAP code for extracting the appraisal status flow

How it works?

Classes description

The class **ZGP_CLS_APPRAISAL_FLOW** encapsulates all the business logic of the appraisal's status flow conversion to the GML format (or DOT format): it reads the DB table containing the status flow, but also the related tables to get additional information on steps / step links (ex.: description of buttons, events…). The constructor of the class accepts as input parameter the appraisal template identifier and his main job is to read the infotype "Status Switch" (Infty 5026). After having read the infotype 5026, it builds a table of flow steps (objects). Each object contains a list of successor.

Data structure used:

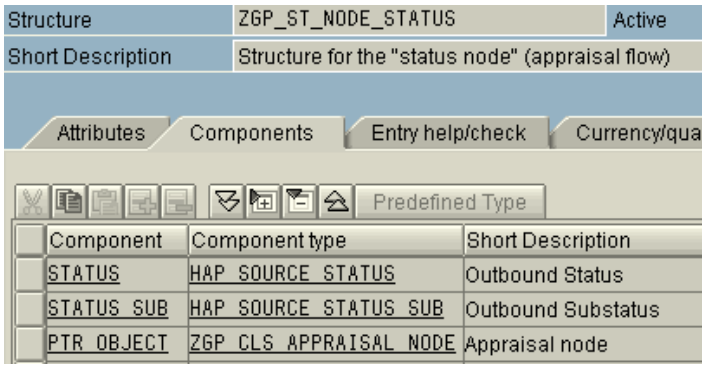


Figure 4 - List of flow steps = table of structures ZGP_ST_NODE_STATUS

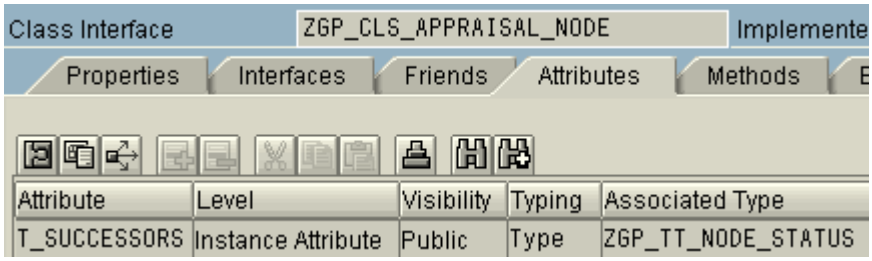


Figure 5 - Each step (object) has a list of successors

The class **ZGP_CLS_APPRAISAL_NODE** represents a flow step. An appraisal flow step is identified by 3 parameters:

	TYPE (Data element)
Template identifier	HAP_TEMPLATE_ID
Status	HAP_SOURCE_STATUS
Sub-status	HAP_SOURCE_STATUS_SUB

These parameters are passed to the class constructor. In some case, an appraisal step needs to access the global flow table. To implement that, I used in the class ZGP_CLS_APPRAISAL_NODE a static pointer initialized in the flow class (ZGP_CLS_APPRAISAL_FLOW) when the infotype 5026 has been loaded. With this technique, only one instance of the flow table resides in memory and is shared by all instances of the class ZGP_CLS_APPRAISAL_NODE; further more, only one initialization is required (outside the class, via the static method SET_T_FLOW of the class ZGP_CLS_APPRAISAL_NODE).

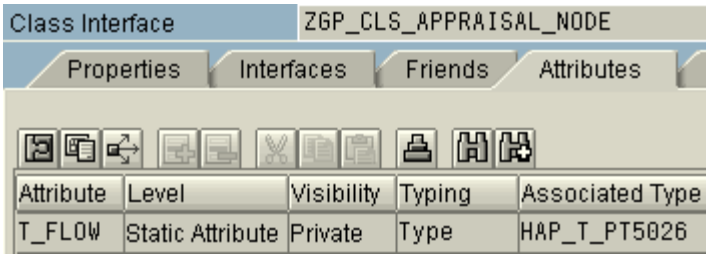


Figure 6 - One instance of table TFLow

The appraisal step class has different methods used to retrieve metadata (that is description of attributes of the class). Example, to get the information stored on the transition the following methods are used:

- GET_EDGE_LABEL Get the label associated to the transition (button + event...)
- GET_EDGE_LABEL_BUTTON Get the name/text of the button that allows the transition
- GET_EDGE_LABEL_EVENT Get the name/text of the event triggered by the transition
- GET_EDGE_LABEL_RULE Get the name/text of the rule that allows the transition

Serialization process (to GML file)

The public method SERIALIZE_TO_GML of the class ZGP_CLS_APPRAISAL_FLOW handles the whole process; that is: it generates a string containing the data in GML format. The string will then be downloaded to the client side. The method generates the main envelope of the GML graph and then call one method to "serialize (in GML format)" the nodes and another method to serialize the edges.

SERIALIZE_TO_GML_NODES: it just loops on all nodes (objects), and calls for each node the method SERIALIZE_TO_GML_NODE. The result of the last method is concatenated with the previous result to get all the nodes serialized in one string. The method SERIALIZE_TO_GML_NODE on the step/node object is just strings manipulations on labels and descriptions of the step to get a GML formatted string.

You have more or less the same process for the edges.

The source code

I provide you the source code as a "nugg" file. This file is used by the SAP Link abap tool: "SAPlink is an open source project that aims to make it easier to share ABAP developments between programmers. It provides the ability to easily distribute and package custom objects". You need to add some plugs to allow the different type of repository objects to be taken in account. More information:

- http://code.google.com/p/saplink

You can easily inspect the source file off-line using a tool that I made: NuggViewer. More information:

http://pierregodart.blogspot.com The NuggViewer program:

NuggViewer.air

Here is the source code:

NUGG_GPI_OSA_STATUS_FLOW.nugg

The report

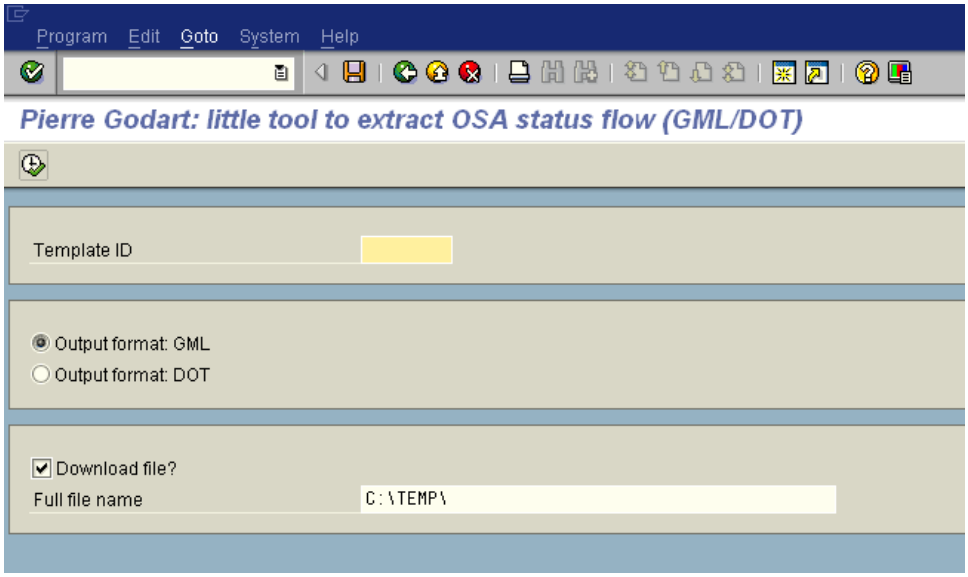


Figure 7 - Abap report

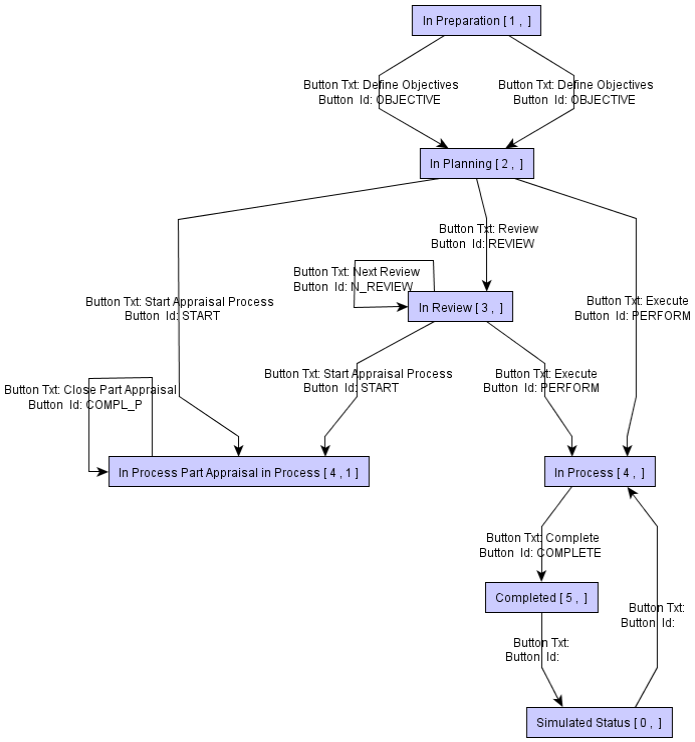
To use the report, just fill the identifier (8 digits) of your template, the output format and the output directory (initialized by default to your temporary directory).

The results

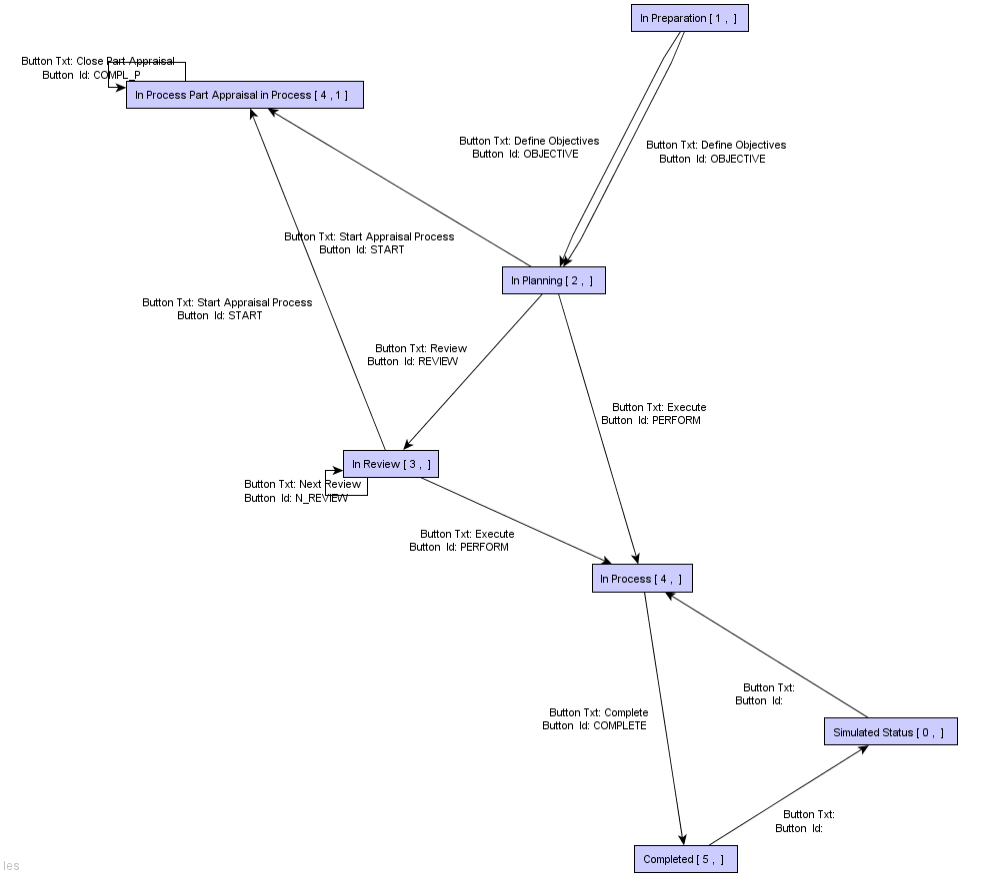
Here are some output pictures rendered using yEd, via importing the files corresponding to the appraisal status flow described above. The different graphics were generated using different pre-configured layout.

After having opened the file, you need to go to the menu and click on:

- Tool -> Fit node to label
- Layout -> Hierarchical -> Classic (for example)



Another example: Layout -> Hierarchical -> Organic



Remark: in theses examples, there are no events raised on button actions.

Possible enhancements

The whole process of rendering can be handled by the program. Indeed, you could code in the Abap source the launch of a command line instruction to open immediately the rendering tool with the generated file as parameter.

Related Content

SAP - HELP

Maurice Hagen's blog on SDN

My LinkedIn Profile

Copyright