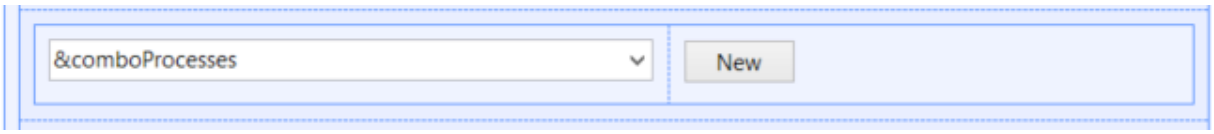
Custom Client Part 2

Neste vídeo veremos a parte 2 do Custom Client. Nesta parte veremos a caixa de entrada e algumas funções como delegar um workitem, completar um workitem e o History.

WORKFLOW INBOX – NEW ACTION

Esta função ocorre quando o usuário seleciona um processo do combo box e então clica no botão New. Isto irá executar uma sub-rotina chamada “New”.




```
104 Sub 'New'
105     If &comboProcesses > 0
106         &processDefinition.Load(&comboProcesses)
107         &processInstance = &processDefinition.CreateInstance()
108         &error = &processDefinition.Error
109         If &error.Code > 0
110             Do 'Error'
111         Else
112             &processInstance.Owner = &user
113             &processInstance.Start()
114             &error = &processInstance.Error
115             If &error.Code > 0
116                 Do 'Error'
117             Endif
118         Endif
119         Commit
120     Endif
121 EndSub

270 Sub 'Error'
271     msg(&error.Message)
272 EndSub
```

Nesta sub-rotina teremos uma variável do tipo de dado Process Definition e executaremos o método Load. Esta função carrega a definição de processo em uma variável, cujas definições de processo recebe do parâmetro selecionado no combo box. Após ter selecionado o tipo de dados de definição de processo, usaremos o método CreateInstance da Definição do Processo. Este método criará uma nova instância de processo e será atribuída à variável de tipo de dados ProcessInstance do workitem. É muito importante verificar se há um erro toda vez que executamos um método para controlar os erros e tomar ações caso haja. Se não houver erro, continuaremos e atribuiremos o usuário que fez login na propriedade Owner da instância do processo. Depois disso, vamos iniciar a instância do processo e, em seguida, começaremos a criar todos os workitems necessários. Posteriormente verificaremos se há algum erro no início deste processo e se existir mostraremos uma mensagem ao usuário.

No final desta sub-rotina está um commit. É importante ter o commit, pois todas as API que suportam o Workflow não fazem commit por si só, então temos que gerenciar a UTL da transação diretamente no commit

WORKFLOW INBOX – EXECUTE ACTION

Agora vamos ver a ação Execute. Essa ação ocorre quando o usuário seleciona um workitem do grid e então pressiona o botão Executar. No evento de mesmo nome, será atribuído o valor "Execute" do domínio WorkflowAction à variável &action do tipo de dado WorkflowAction. Em seguida, executará a sub-rotina Button Pressed, onde a variável workitem, que é o tipo de dado workflow workitem, usará o método Load para poder carregar o workitem selecionado na variável Id, obtido do grid.

```
38 Event 'Execute'
39       &action = WorkflowAction.EXECUTE
40       Do 'Button Pressed'
41 EndEvent
42
```

```
78 Sub 'Button Pressed'
79       &workitem.Load(&id)
80       Do Case
81            Case &action = WorkflowAction.EXECUTE
82                 Do 'Execute'
83            Case &action = WorkflowAction.COMPLETE
84                 Do 'Complete'
85            Case &action = WorkflowAction.PREVIEW
86                 Do 'Preview'
87            Case &action = WorkflowAction.DELEGATE
88                 Do 'Delegate'
89            Case &action = WorkflowAction.COLLABORATE
90                 Do 'Collaborate'
91            Case &action = WorkflowAction.VIEW_HISTORY
92                 Do 'View History'
93            Case &action = WorkflowAction.ADD_COMMENTS
94                 Do 'Add Comments'
95            Case &action = WorkflowAction.VIEW_DOCUMENTS
96                 Do 'Documents'
97       EndCase
98    Commit
99    If &error.Code > 0
100       Do 'Error'
101   Endif
102 EndSub
```

Depois que o workitem for carregado na variável, será executada a sub-rotina chamada Execute. Na sub-rotina Execute, a primeira coisa que deve fazer é definir o In- process state do workitem, pelo que executará a sub-rotina "Set In-process state", que chama a sub-rotina Take. Nesta sub-rotina será verificado o status do workitem. Se o workitem está OPEN_ACTIVE_READY, significa que não foi atribuído a nenhum usuário, a primeira coisa que deve fazer é atribuir o workitem ao usuário que executou a caixa de entrada. Então usamos o método "Assign" da variável workitem, passando a variável &user do tipo Workflowuser que foi carregada no evento Start. Se não houver erros, continuará.

Caso o workitem esteja OPEN_ACTIVE_ASSIGNED, validará que o usuário logado é o usuário que vem executando este workitem, mas caso o usuário que executa não seja o usuário que executou anteriormente o workitem, irá reatribuir o workitem ao usuário que está executando a caixa de entrada, então aqui usaremos o método Reassign do workitem e reatribuiremos ao participante o usuário que está executando a caixa de entrada conforme mostrado no código usando a variável workitem com o método participante e a variável user do tipo WorkflowUser..

```
155 Sub 'Execute'
156     Do 'Set In-Process State'
157     If &error.Code = 0
158         &app = &workitem.Activity.Application
159         If Not &app.IsEmpty()
160             Do 'Open App'
161         Else
162             Do 'Documents'
163         Endif
164     Endif
165 EndSub
```

```
Sub 'Set In-Process State'

    Do 'Take'
    If &error.Code = 0
        &workitem.ChangeState(WorkflowWorkitemState.OPEN_ACTIVE_INPROCESS)
        &error = &workitem.Error
    Endif

EndSub
```

```
145 Sub 'Take'
146     Do Case
147         Case &workitem.State = WorkflowWorkitemState.OPEN_ACTIVE_READY
148             &workitem.Assign(&user)
149             &error = &workitem.Error
150         Case &workitem.State = WorkflowWorkitemState.OPEN_ACTIVE_ASSIGNED
151             If &workitem.Participant.Id = &user.Id
152                 //Do nothing
153             Else
154                 If &workitem.Participant.Id = !'N/A' //Assigned to a role or a list of users
155                     &workitem.Reassign(&workitem.Participant, &user)
156                     &error = &workitem.Error
157                 Else
158                     &error.Code = 203 //The task is already assigned to another user
159                 Endif
160             Endif
161
162         Otherwise
163             &error.Code = 200 //Invalid transition
164     EndCase
165 EndSub
```

Depois do método “Take”, mudará o estado do workitem usando o método “ChangeState” e mudará o estado para OPEN_ACTIVE_INPROCESS.

Uma vez que que o workitem tenha o status correto, será aberta a aplicação definida no diagrama da definição do processo. Nesse diagrama, o usuário indica na propriedade application a atividade que será executada no workitem. Então vamos usar novamente a variável workitem, activity e a propriedade application, para obter a aplicação que será executada.

Se a aplicação não estiver vazia, executará a sub-rotina Open App. Esta sub-rotina tem uma variável do tipo de dado Window para criar uma janela pop-up e abrir a aplicação.

```
Sub  'Open App'
     If Not &app.IsEmpty()
          &app = WorkflowBuildApplicationUrl(&app, &workitem)
          &window.Url = &app
          &window.Autoresize = False
          &window.Width = WorkflowWindowSize.APP_WIDTH
          &window.Height = WorkflowWindowSize.APP_HEIGHT
          &window.Open()
     Endif
EndSub
```

Caso a variável application esteja vazia, executará a sub-rotina “Documents”. Neste caso, quando não houver uma aplicação associada a uma atividade de processo, os documentos aparecerão em um Trabalhar com documentos, caso tenha sido definido para isso. Portanto, ele validará que a atividade do workitem tem a propriedade canWorkWithtDocuments como True, se não for verdade, validará se o status do workitem está correto. Se estiver correto, executará o Web panel Work With Document para mostrar o trabalho com documentos. Se não pode trabalhar com os documentos configurados na atividade, não fará nada porque não tem uma aplicação e não pode trabalhar com os documentos.

```
Sub 'Documents'
     If &workitem.Activity.canWorkWithDocuments = True
          If &workitem.State <> WorkflowWorkitemState.OPEN_ACTIVE_INPROCESS
               Do 'Set In-Process State'
          Endif
          If &error.Code = 0
               &window.Object = WorkflowWorkWithDocuments.Create(&workitem.Id)
               &window.Open()
          Endif
     Else
          msg('Operation not allowed')
     Endif
EndSub
```

A seguir veremos a ação complete.

WORKFLOW INBOX COMPLETE ACTION

Quando um usuário pega um workitem do grid e clica no botão Complete, executará o evento de mesmo nome. Aqui usaremos o domínio WorkflowAction com o valor complete e atribuído à variável action. Depois disso, executará a sub-rotina “Button Pressed”, assim como na ação Execute será carregado o Id do Workitem selecionado na variável do tipo de dado Workflow Workitem.

```
43  Event 'Complete'
44      &action = WorkflowAction.COMPLETE
45      Do 'Button Pressed'
46  EndEvent
47
```

```
78  Sub 'Button Pressed'
79          &workitem.Load(&id)
80          Do Case
81              Case &action = WorkflowAction.EXECUTE
82                  Do 'Execute'
83              Case &action = WorkflowAction.COMPLETE
84                  Do 'Complete'
85              Case &action = WorkflowAction.PREVIEW
86                  Do 'Preview'
87              Case &action = WorkflowAction.DELEGATE
88                  Do 'Delegate'
89              Case &action = WorkflowAction.COLLABORATE
90                  Do 'Collaborate'
91              Case &action = WorkflowAction.VIEW_HISTORY
92                  Do 'View History'
93              Case &action = WorkflowAction.ADD_COMMENTS
94                  Do 'Add Comments'
95              Case &action = WorkflowAction.VIEW_DOCUMENTS
96                  Do 'Documents'
97          EndCase
98      Commit
99      If &error.Code > 0
100         Do 'Error'
101     Endif
102 EndSub
```

Depois disso, executará a sub-rotina Complete e nesta sub-rotina verificará se o workitem está com o status correto antes de continuar. Todos os workitems a completar devem ter o status OPEN_ACTIVE_INPROCESES, o que significa que foram executados previamente.

Se tiver o status correto, completará o workitem, o que significa que este workitem será encerrado e serão criados os seguintes workitems do processo. Se houver um erro, por exemplo, porque o usuário deve selecionar um caminho opcional, será solicitado selecionar uma atividade e escolher a rota que deve ser seguida. Caso o erro seja devido ao fato de ser um processo Ad-hoc e deve ser selecionada também a próxima atividade a ser executada, mas se o caso for que o erro seja devido ao fato de que são necessários alguns comentários nesta atividade, será aberto um prompt do Workflow, para que o usuário insira os comentários necessários para completar esta tarefa. Se houver um erro, será mostrado ao usuário, se não houver, a tarefa finalizou.

```
Sub 'Complete'
    If &workitem.State = WorkflowWorkitemState.OPEN_ACTIVE_INPROCESS
        &workitem.Complete()
        &error = &workitem.Error
        If &error.Code > 0
            Do Case
                Case &error.Code = WorkflowError.OPTIONALS_SELECTION_REQUIRED
                    &window.Object = WorkflowSelectActivity.Create(&workitem.Id, WorkflowSelectionMode.OPTIONALS, WorkflowAction.COMPLETE)
                    &window.Open()
                    &error = new()

                Case &error.Code = WorkflowError.ADHOC_SELECTION_REQUIRED
                    &window.Object = WorkflowSelectActivity.Create(&workitem.Id, WorkflowSelectionMode.ADHOC, WorkflowAction.COMPLETE)
                    &window.Open()
                    &error = new()

                Case &error.Code = WorkflowError.COMMENTS_REQUIRED
                    &window.Object = WorkflowComments.Create(&workitem.Id, WorkflowObjectType.WORKITEM, False)
                    &window.Open()
                    &error = new()

                Otherwise
                    Do 'Error'
            EndCase
        Endif
    Else
        &error.Code = 204  // The task has not been processed yet
    Endif
EndSub
```

WORKFLOW INBOX – HISTORY ACTION

Na caixa de entrada do Custom Client com o workitem pré-selecionado no grid e pressionando o botão History, será exibido o History Panel. Este panel apresenta uma lista de workitems que foram executados a partir da instância do processo. Para isso, será executado o seguinte código do evento History, e este evento utilizará, como as demais ações deste panel, atribuir na variável action o domínio WorkflowAction com o valor VIEW_HISTORY e então executar a sub-rotina Button Pressed. Nesta sub-rotina será carregada a variável workitem, com o Id que foi selecionado no grid. Este é o Id de workitem e então temos o workitem atribuído à variável.

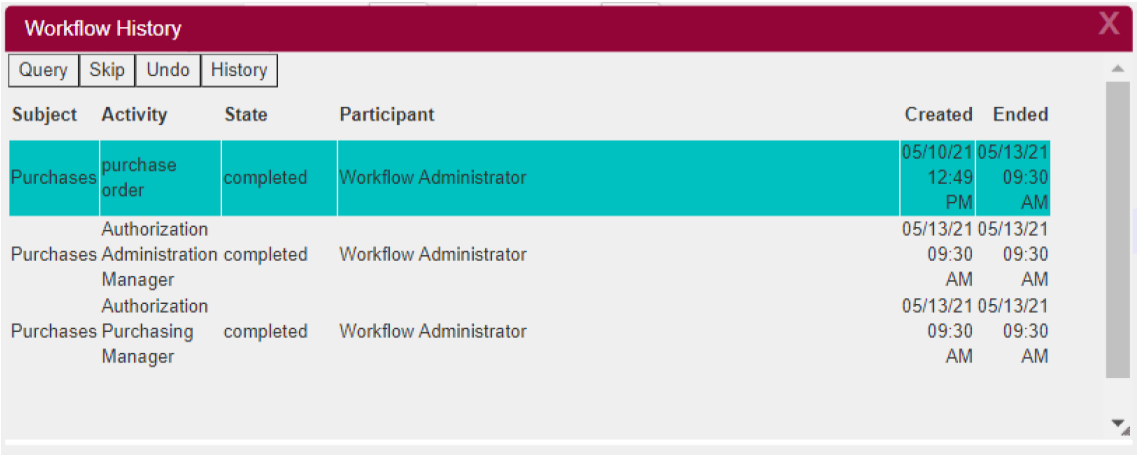
```
Event 'History'
     &action = WorkflowAction.VIEW_HISTORY
     Do 'Button Pressed'
EndEvent
```

```
Sub 'Button Pressed'
          &workitem.Load(&id)
          Do Case
               Case &action = WorkflowAction.EXECUTE
                    Do 'Execute'
               Case &action = WorkflowAction.COMPLETE
                    Do 'Complete'
               Case &action = WorkflowAction.PREVIEW
                    Do 'Preview'
               Case &action = WorkflowAction.DELEGATE
                    Do 'Delegate'
               Case &action = WorkflowAction.COLLABORATE
                    Do 'Collaborate'
               Case &action = WorkflowAction.VIEW_HISTORY
                    Do 'View History'
               Case &action = WorkflowAction.ADD_COMMENTS
                    Do 'Add Comments'
               Case &action = WorkflowAction.VIEW_DOCUMENTS
                    Do 'Documents'
          EndCase
     Commit
     If &error.Code > 0
          Do 'Error'
     Endif
EndSub
```

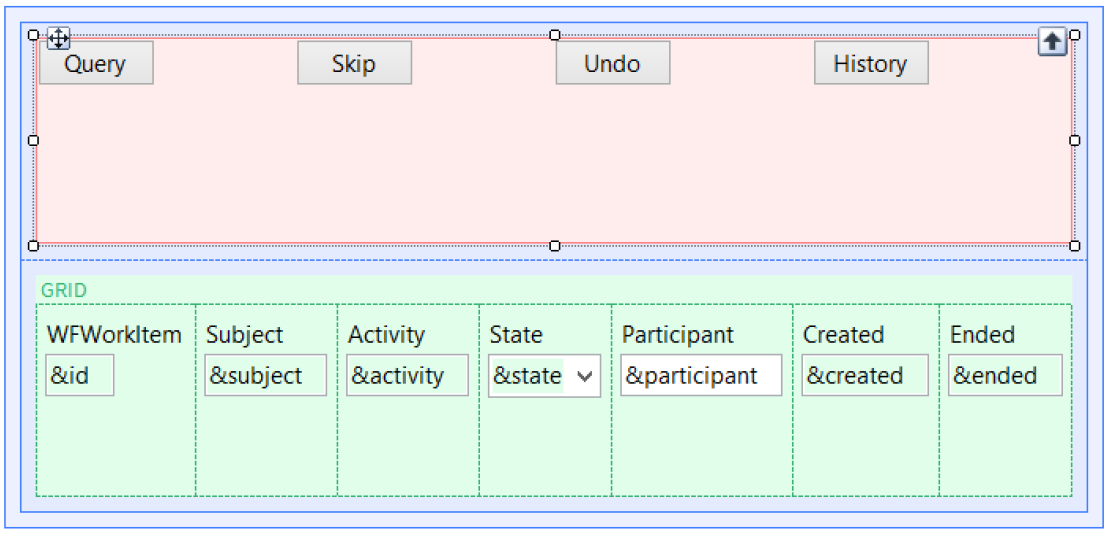
Depois disso, executaremos a sub-rotina View History. Nesta sub-rotina, será usada a variável Window para abrir um Web component chamado Workflow History. Este panel recebe por parâmetro o Id da instância do processo usando a variável workitem que foi atribuída na sub-rotina Button pressed. Esta variável tem uma propriedade chamada ProcessInstanceId, que pega a instância do processo deste workitem e então abrirá a página com a propriedade Open da variável Window.

```
Sub 'View History'
     &window.Object = WorkflowHistory.Create(&workitem.ProcessInstanceId)
     &window.Open()
EndSub
```

Se observamos a execução deste panel, veremos que existe uma lista de todos os workitems que foram executados desta instância de processo, onde é mostrado o assunto, a atividade, o status, o participante, quando foi criado, quando terminou e uma lista de ações que podem ser executadas para estes workitems.



Se observamos o design do Web panel, ele possui um grid com todos os elementos que mencionamos e uma tabela com todas as ações.



Se observarmos o código, veremos na seção regras que recebe o parâmetro &ProcessInstanceId.

No Evento Start fará o mesmo que todos os painéis no Custom Client, verificará se existe uma sessão válida e então atribuirá o usuário conectado, que obtivemos pela propriedade de ConnectedUser da variável server do tipo WorkflowServer. Depois disso, será carregada na variável &ProcessInstance, que é do tipo de dados WorkflowProcessInstance, o Id de instância do processo que recebeu por parâmetro. Posteriormente no evento Refresh, obterá todos os workitems que foram criados para esta instância, utilizando a propriedade workitems.

Esta propriedade retorna uma lista de workitems, são do tipo de dado WorkflowWorkitems e são atribuídos à variável &workitem. Esta lista será iterada no evento de carga e para cada workitem desta lista, atribuirá seus valores ao grid. Por exemplo, atribuirá o Id do workitem, o nome da atividade, o status, o assunto da instância do processo, o nome do participante que executou esta tarefa, a data em que foi criada e a data em que terminou.

```
Event Start
     &server = WorkflowCheckServerSession()
     &user = &server.ConnectedUser
     &processInstance.Load(&processInstanceId)
EndEvent

Event Grid.Refresh
     &workitems = &processInstance.Workitems
EndEvent

Event Grid.Load
     For &workitem in &workitems
          &id            = &workitem.Id
          &activity      = &workitem.Activity.Name
          &state         = WorkflowWorkitemState.Convert(&workitem.State)
          &subject       = &workitem.ProcessInstance.Subject
          &participant   = &workitem.Participant.Name
          &created       = &workitem.Created
          &ended              = &workitem.Ended

          Grid.Load()
     Endfor
EndEvent
```

Quando executarmos esta ação, vamos delegar um workitem que temos em nossa caixa de entrada para outra pessoa. Para isso, no evento Delegate, assim como em outras ações, atribuiremos WorkflowAction, mas com o valor DELEGATE na variável &action e então será executada a sub-rotina "Button Pressed". Nesta sub-rotina, carregará o Id dos workitems selecionados no grid para a variável workitem e executará a sub-rotina Delegate. Nesta sub-rotina vamos verificar se a atividade do workitem tem habilitada a delegação de funções. Você pode habilitar a delegação selecionando no diagrama a atividade da tarefa e nas propriedades colocando true na propriedade "Allow delegation", isto habilitará a ação de delegação. Se a tarefa puder delegar, apresentará o fluxo de trabalho do panel atribuído ao usuário. Este panel apresentará todos os usuários que podem receber esta tarefa.

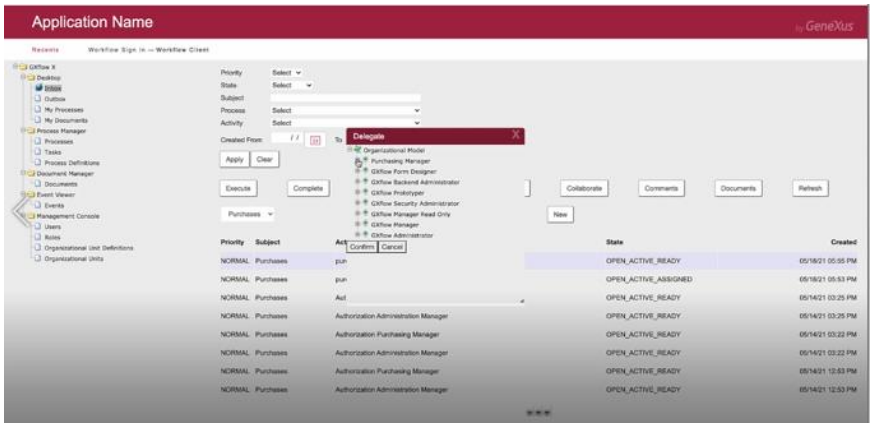
```
Event 'Delegate'
     &action = WorkflowAction.DELEGATE
     Do 'Button Pressed'
EndEvent
```

```
Sub 'Button Pressed'
     &workitem.Load(&id)
     Do Case
          Case &action = WorkflowAction.EXECUTE
               Do 'Execute'
          Case &action = WorkflowAction.COMPLETE
               Do 'Complete'
          Case &action = WorkflowAction.PREVIEW
               Do 'Preview'
          Case &action = WorkflowAction.DELEGATE
               Do 'Delegate'
          Case &action = WorkflowAction.COLLABORATE
               Do 'Collaborate'
          Case &action = WorkflowAction.VIEW_HISTORY
               Do 'View History'
          Case &action = WorkflowAction.ADD_COMMENTS
               Do 'Add Comments'
          Case &action = WorkflowAction.VIEW_DOCUMENTS
               Do 'Documents'
     EndCase
     Commit
     If &error.Code > 0
          Do 'Error'
     Endif
EndSub
```

```
Sub 'Delegate'
     If &workitem.Activity.canDelegate = True
          &window.Object = WorkflowAssign.Create(&workitem.Id, WorkflowAction.DELEGATE)
          &window.Open()
     Else
          msg('Operation not allowed')
     Endif
EndSub
```

Se observarmos a função de delegar em execução, podemos ver que se selecionarmos nosso workitem do grid da caixa de entrada de nosso fluxo de trabalho e pressionarmos o botão delegar, aparecerá uma página web com todo o modelo organizacional e todas as roles, que pode ter a Knowledge base, é possível abrir uma função, visualizar os usuários e selecionar um, confirmar e esta tarefa será delegada a esse usuário.

Este Web panel é criado com um objeto Window e recebe dois parâmetros, o Id workitem, que foi carregado na sub-rotina de Button Pressed, e a ação, neste caso Workflow Action delegate.



No layout veremos que é diferente do resto dos painéis, tem três controles e dois botões, confirmar e cancelar.

Se vemos o código, veremos que recebe dois parâmetros, Id do workitem e action. No evento Start ele irá verificar a sessão de fluxo de trabalho com o procedimento WorkflowCheckServerSession igual a outros painéis. Em seguida, obterá do WorkflowServer que foi carregado o Modelo Organizacional, que faz parte do servidor. Com o método Organizational Model isto retornará uma variável do tipo de dados WorkflowOrganizationalModel, que neste caso será a variável OrgModel, então carregará o parâmetro de recepção de workitems para a variável workitem

.

Então é feito Refresh e é invocada a sub-rotina “Populate Tree”. Esta sub-rotina lista todas as roles usando o método “list roles”, este recebe um parâmetro de filtro, neste caso está vazio, e receberá aqui todos as roles do Modelo Organizacional. Carregará o caminho de todas as roles da árvore e para cada função da lista também carregará as informações associadas.

```
127 Sub 'Populate Tree'
128
129     &roles = &orgModel.ListRoles(&filter)
130
131     &root.Id = !"OM"
132     &root.Name = "Organizational Model"
133     &root.Icon = WorkflowOrganizationalModel.Link()
134     &root.IconWhenSelected = WorkflowOrganizationalModel.Link()
135     &root.Expanded = True
136     &treeNodeCollectionData.Add(&root)
137
138     For &role in &roles
139
140         If &role.hasParent = False
141
142             &treeNode = new()
143             &treeNode.Id = Trim(Str(&role.Id))
144             &treeNode.Name = &role.Name
145             &treeNode.Icon = WorkflowRole.Link()
146             &treeNode.IconWhenSelected = WorkflowRole.Link()
147             &treeNode.DynamicLoad = True
148             &root.Nodes.Add(&treeNode)
149
150         Endif
151
152     Endfor
153
154 EndSub
```

WORKFLOW ASSIGN – WEB PANEL CODE

Quando está sendo atribuído um item porque selecionamos um usuário ou função e, em seguida, pressionamos confirmar, validará que o Nó da Árvore não está vazio e a ação não é colaborar, obterá a função selecionada na árvore com o ID selecionado e a variável, então obterá a função, no Modelo Organizacional, obterá seu ID. Em seguida, obteremos a função de fluxo de trabalho na variável Role, se não houver erro, será atribuído. Caso estejamos atribuindo a um usuário, devemos obtê-lo através do Modelo Organizacional usando GetUserById, e então, se não houver erro, validaremos a ação. Caso a ação seja delegada, usaremos a variável Workitem.DELEGATE, este método irá delegar este workitem ao referido usuário. Se não houver erro, é feito commit para salvar todas as alterações e depois retornar à página web.

```
Event Enter
    If Not &selectedTreeNode.Id.IsEmpty()
        If &selectedTreeNode.Id.ToNumeric() > 0 //Role
            If &action <> WorkflowAction.COLLABORATE
                &role = &orgModel.GetRoleById(&selectedTreeNode.Id.ToNumeric())
                &error = &orgModel.Error
                If &error.Code = 0
                    If Block
                    &error = &workitem.Error
                Endif
            Endif
        Else // User
            &user = &orgModel.GetUserById(&selectedTreeNode.Id)
            &error = &orgModel.Error
            If &error.Code = 0
                Do Case
                    Case &action = WorkflowAction.COLLABORATE
                        &workitem.Collaborate(&user)
                    Case &action = WorkflowAction.DELEGATE
                        &workitem.Delegate(&user)
                    Case &action = WorkflowAction.REASSIGN
                        &workitem.Reassign(&workitem.Participant, &user)
                EndCase
                &error = &workitem.Error
            Endif
        Endif
        Commit
        If &error.Code = 0
            Return
        Else
            Do 'Error'
        Endif
    Endif
EndEvent
```

Neste vídeo vimos o uso do Custom Client, como podemos importá-lo em uma Knowledge base, porque o utilizamos e algumas de suas ações.