A Nouveau Systems White Paper

www.nouveausystems.com

A Call Center Framework Using Nouveau Alliance

By Larry Smith, Co-founder, Nouveau Systems

February, 2008

Contents

Introduction

This white paper describes a turn-key framework that supports the creation and execution of a call center.

Problem Statement

The goal of this white paper is to build a call center for a consumer rating organization who surveys customers of businesses to determine their satisfaction. The call center is staffed with people at multiple call stations. A call center server directs calls to a call station and steps the person conducting the survey through a series of questions. At the end of the call, the results are recorded in a database.

Previous Options

Currently, the only options available to a survey taker are to purchase a turn-key call center system from a Phone Dialer vendor or to create a custom software solution. In the case of the Phone Dialer system, the delivered product is large and not easily customizable. The schema is all-encompassing, containing extraneous fields for the typical survey.

Our Solution

This paper presents an easily customizable Call Center Framework composed of Nouveau Alliance Workflows. Using this framework, you need to configure a single database table to accumulate the call center results and a single workflow to provide the flow, interaction and access to the database table. The framework provides the support for the management of the call center.

Powerful Modeling

Nouveau Systems' FlowSpace provides a library of primitives that allow you to coordinate your call center in a simple, yet powerful way. If some customization is needed, you can add to or modify these

primitives via Java or a myriad of supported scripting languages. When complete, these customizations can be added to the library of built-in primitives to manage new call centers.

Integrated RDMS Database

You can store and retrieve relational data from a full-featured, fully integrated RDBMS. Using the database access primitives of FlowSpace, you can seamlessly integrate the database access in your call center.

Web-based Servlet Interface

You can enter your application logic graphically by defining workflows with FlowSpace. You then can use a Workflow Servlet to create a web application. This enables domain experts to define and modify business logic without extensive knowledge of web programming.

Implementation

The implementation of the Call Center Framework is best described by its Process, Database, and Workflow models.

Process Model

The Call Center Framework has a client process to interact with the survey taker (caller) and a server to process the caller's requests for customer contacts.

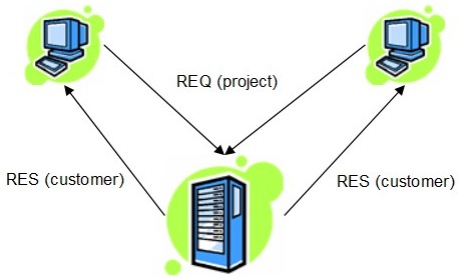

Figure 1 – Client/Server Communication

The Call Center Client processes the caller login, including the selection of the Survey Project (the survey classification: "Survey of All Automotive Repair Firms", "Survey of Joe's Eatery"). It then posts a REQUEST (to the database) with the caller's ID and the desired project to The Call Center Server and waits for a RESPONSE.

The Call Center Server queries its database of target customers to be surveyed, opens a Survey Call to track the REQUEST and responds with the selected customer to the requesting caller. The server returns to a loop handling requests from multiple callers.

The Call Center Client receives the RESPONSE, conducts the survey, updates the Survey Call and creates a Completed Survey with the information gathers in the survey. It then requests another customer to survey.

In this system, the caller manually dials the customer when the number appears in the client window. In a more advanced system, the server interacts with special phone dialer hardware that calls the customer and only transfers the connection to the caller if the customer answers.

Database Model

The persistence of the Call Center Framework is provided by the embedded use of Derby/DB RDBMS in Nouveau Alliance. The database tables can be categorized by use.

Client/Server Communication

The following tables are used to facilitate the "handshaking" between the Call Center Client and Server, handling the call requests and responses. In addition, a Process Interrupt table is provided to terminate the client and server processes as needed.

CallRequest

This table is used by clients, on behalf of a caller, to request a call for a given project. The client creates a row and the server acknowledges the request (by querying and deleting the row) and generates a corresponding response.

RequestID (INTEGER, GENERATED)
: The unique identifier of the CallRequest used as a primary key.

CallerID (INTEGER)
: The identifier of the caller making the call request.

ProjectID (INTEGER)
: The identifier of the desired survey project.

CallResponse

This table is used by servers to respond to client caller requests for customer calls for a project specified survey. The server creates a row corresponding to the request tagged with the caller's ID and the ID of the customer to be surveyed.

ResponseID (INTEGER, GENERATED)
: The unique identifier of the CallResponse used as a primary key.

CallerID (INTEGER)
: The identifier of the caller who made the request.

CustomerID (INTEGER)
: The identifier of the customer to be surveyed.

A "0" customer ID is returned by the server to signify that all customers that can be surveyed at the current time for a given survey have been attempted.

ProcessInterrupts

This table is used to register interrupt requests for client and server processes. The client user, or the server administrator, adds rows to this table to interrupt the loops of the respective process. What happens after the interrupt depends on the workflow of the process.

ProcessID (INTEGER, GENERATED)
: The unique identifier of the ProcessInterrupt used as a primary key.

keyID (INTEGER)
: The identifier of the process to be stopped (0 = server and # = client callerID)

Survey Management

The following table is used the track the attempts to conduct the survey, relating all of the constituent components of the survey.

SurveyCall

This table is used to document each call that is made to a customer for a given survey. The server makes a call for every request made by a caller. This row is subsequently updated when its result code is known.

CallID (INTEGER, GENERATED)
: The unique identifier of the SurveyCall used as a primary key.

ProjectID (INTEGER)
: The identifier of the project associated with the call.

TargetID (INTEGER)
: The identifier of the target vendor whose products and services are to be surveyed.

CustomerID (INTEGER)
: The identifier of the target vendor's customer to be surveyed.

CallerID (INTEGER)
: The identifier of the survey taker (caller).

CallTimeStamp (TIMESTAMP)
: The time of the call.

Station (VARCHAR(32))
: The station name of the caller.

ResultCode (CHAR(3))
: The code representing the resolution of the call (e.g., 'CBL' for 'Call Back Later', 'WRG' for 'Wrong Number' …). When a call has been completed, the return code is 'CMP'

and when a call is 'in-progress', the result code is NULL.

Survey Input Data

The following tables provide the survey-specific data needed to initiate a survey.

SurveyProject

This table represents a survey to be taken. It has a title for documentation purposes.

ProjectID (INTEGER, GENERATED)
: The unique identifier of the SurveyProject used as a primary key.

Title (VARCHAR(80))
: The title of the project.

TargetID (INTEGER)
: The identifier of a survey target included in the project.

SurveyTarget

This table is used to specify the vendors that are to be surveyed.

TargetID (INTEGER, GENERATED)
: The unique identifier of the SurveyTarget used as a primary key.

Name (VARCHAR(50))
: The name of the target customer being surveyed.

BusinessType (VARCHAR(50))
: The name of the business category of the target customer (e.g., "Auto Repair", Florist",…).

SurveyCustomer

This table is used to identify the customers to be called for the target vendor to be surveyed.

CustomerID (INTEGER, GENERATED)
: The unique identifier of the CallRequest used as a primary key.

Name (VARCHAR(200))
: The name of the vendor's customer being called.

Phone (VARCHAR(40))
: The phone number of the vendor's customer being called.

TargetID (INTEGER)
: The identifier of the target customer.

SurveyCaller

This table is used to specify the available survey takers (callers) that conduct the surveys.

CallerID (INTEGER, GENERATED)
: The unique identifier of the SurveyCaller used as a primary key.

Name (VARCHAR(50))
: The first and last name of the survey taker.

Survey Output Data

The following table is used to capture the completed results of the survey.

CompletedSurvey

This table represents the completed survey information. In this case, it contains satisfaction information corresponding to the target vendor associated with the specified call.

CompletedSurveyID (INTEGER, GENERATED)
: The unique identifier of the CompletedSurvey used as a primary key.

CallID (INTEGER)
: The identifier of the call that generated the completed survey.

Satisfaction (INTEGER)
: The level of satisfaction, low to high, represented by a value of 1 to 10.

Loyalty (INTEGER)
: An indicator of whether the customer will use the services of the target vendor again (1 for Yes and 0 for No).

Dissatisfaction (VARCHAR(500))
: An explanation as to why the customer rated the vendor so low (a satisfaction level of 5 or less).

Workflow Model

The Call Center Framework consists of three main workflows representing the client and server processes and the actual survey sequence.

Call Center Client Workflow

The Call Center Client workflow is responsible for managing the survey taker (caller) session. It handles the caller login, request for a customer (a loop), processes

the response, conducts the survey and gets the next customer (loop). The following is an "overview" diagram of the workflow.

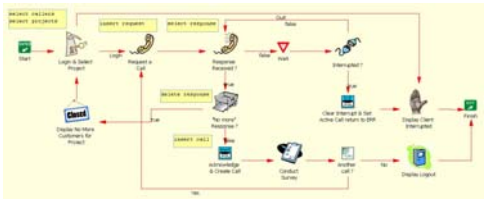

Figure 2 – Call Center Client Workflow

For a better understanding, the workflow has been broken into two parts, with a listing of its constituent workflow nodes, activities and properties.

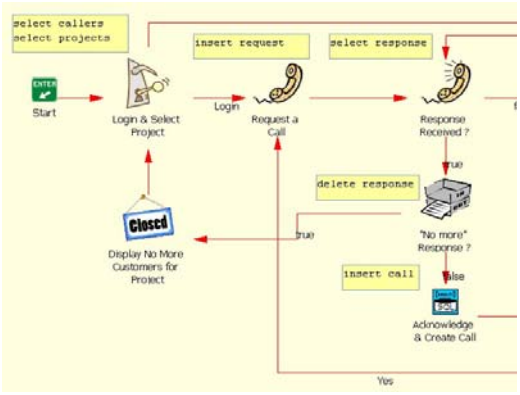

Figure 3 – Call Center Client Workflow (front)

Login & Select Project

This node SELECTS a list of callers and projects from the database, presents "combo box" interfaces for both, letting the user select one of each to set corresponding INTERPRETER properties.

SetPropertyActivityImpl
  INTERPRETER.selectStatement=SELECT name,callerID from SurveyCaller
  INTERPRETER.selectProps=INTERPRETER.namex,INTERPRETER.callerID
  INTERPRETER.resultsRow='<option value="$INTERPRETER.callerID">$INTERPRETER.namex</option>'
  INTERPRETER.results=""

SubWorkflowActivityImpl
  workflowOID=(getValues workflow OID)

SetPropertyActivityImpl
  IINTERPRETER.station=myhost
  INTERPRETER.nameOptions=$INTERPRETER.results

SetPropertyActivityImpl
  INTERPRETER.selectStatement=SELECT title,projectID from SurveyProject
  INTERPRETER.selectProps=INTERPRETER.title, INTERPRETER.projectID
  INTERPRETER.resultsRow='<option value="$INTERPRETER.projectID">$INTERPRETER.title</option>'
  INTERPRETER.results=""

SubWorkflowActivityImpl
  workflowOID=(getValues workflow OID)

SetPropertyActivityImpl
  INTERPRETER.surveyOptions=$INTERPRETER.results

SetPropertyFromTextFileActivityImpl
  URLspecifier=$$INSTANCE.website/doc/examples/CallCenter/templates/login_template.html
  targetPropertyName=unformattedPage

SetPropertyActivityImpl
  formattedPage="$$unformattedPage"
  SERVLET.RESPONSE.print=$formattedPage
  close=$SERVLET.RESPONSE.close

WaitActivityImpl
  waitTime=NULL

SetPropertyActivityImpl
  TARGET=$INSTANCE.answer

Note: From this point on the parameter for the Wait is the same (unless otherwise noted). It will be abbreviated by "…" to simplify the presentation.

Request a Call

This node INSERTS a request for a customer in the database.

SetPropertyActivityImpl
  INTERPRETER.callerID=$INSTANCE.callerID
  INTERPRETER.projectID=$INSTANCE.projectID

DBConnectActivityImpl
  driver=org.apache.derby.jdbc.EmbeddedDriver
  connectionProperty=INSTANCE.DB_CONNECTION
  connectionURL=jdbc:derby:'$WORKSPACE'/defaultDB;create = true
  connectionUser=APP
  connectionPassword=APP

DBExecuteActivityImpl
  executeString=INSERT INTO CallRequest (callerID,projectID) VALUES ($INTERPRETER.callerID,$INTERPRETER.projectID)
  connectionProperty=$INSTANCE.DB_CONNECTION
  returnStatusProperty=INTERPRETER.DB_RETURN_STATUS

DBCloseActivityImpl
  connectionProperty=$INSTANCE.DB_CONNECTION

Note: From this point on the parameterization for the database connect and close activities is the same. They will be abbreviated by "…" to simplify the presentation.

Response Received ?

This node SELECTS a response from the Survey Server to the request. A return status is used to process the response or wait until one is returned.

DBConnectActivityImpl …

DBExecuteQueryActivityImpl
executeString=SELECT responseID,customerID FROM CallResponse WHERE callerID=$INTERPRETER.callerID
…
propertyNames=INTERPRETER.responseID,INTERPRETER.customerID
returnStatusProperty=TARGET

DBCloseActivityImpl …

"No more" Response ?

This node DELETES the response from the database and checks to see if the returned customer ID is 0 (signifying no more customers are currently available for the given survey project.

DBConnectActivityImpl …

DBExecuteActivityImpl
executeString=DELETE FROM CallResponse where responseID = $INTERPRETER.responseID
…

DBCloseActivityImpl …

CompareActivityImpl
value1=0
value2=$INTERPRETER.customerID
comparison=EQUAL
trueLabel=true
falseLabel=false

Display No More Customers for Project

This node informs the caller that there are no more customers to survey for the current project.

InformActivityImpl
message=No more customers to be surveyed today for the current project.

SetPropertyFromTextFileActivityImpl
URLspecifier=$$INSTANCE.website/doc/examples/CallCenter/templates/info_template.html
targetPropertyName=unformattedContents

SetPropertyActivityImpl
formattedContents="$$unformattedContents"
SERVLET.RESPONSE.print=$formattedContents

WaitActivityImpl …

Acknowledge & Create Call

This node DELETES the response from the database, creates a SurveyCall object to track the survey and re-fetches the called corresponding to the created call.

DBConnectActivityImpl …

DBExecuteActivityImpl
executeString=DELETE FROM CallResponse where responseID = $INTERPRETER.responseID
…

DBExecuteQueryActivityImpl
executeString=SELECT targetID FROM SurveyProject WHERE projectID = $INTERPRETER.projectID…

DBResultSetActivityImpl
propertyNames=INTERPRETER.targetID
…

DBExecuteActivityImpl
executeString="INSERT INTO SurveyCall (projectID,targetID,customerID,callerID,callTimeStamp,station) VALUES ($INTERPRETER.projectID,$INTERPRETER.targetID,$INTERPRETER.customerID,$INTERPRETER.callerID,CURRENT_TIMESTAMP,'$INTERPRETER.station')"
…

DBExecuteQueryActivityImpl
executeString=SELECT callID from SurveyCall WHERE projectID=$INTERPRETER.projectID AND targetID=$INTERPRETER.targetID AND customerID=$INTERPRETER.customerID AND callerID=$INTERPRETER.callerID AND resultCode IS NULL
…

DBResultSetActivityImpl
propertyNames=INTERPRETER.callID
…

DBCloseActivityImpl …

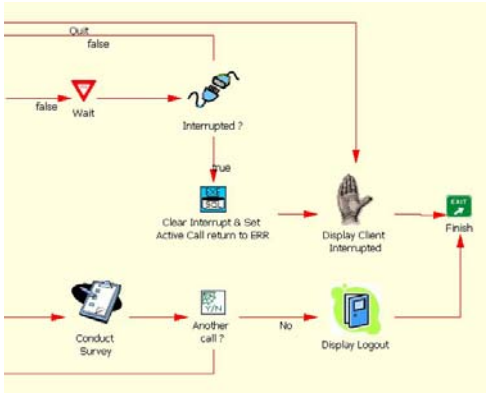

Figure 4 – Call Center Client Workflow (back)

Conduct Survey

This node invokes the sub-workflow "Survey" described later in this section.

SubWorkflowActivityImpl
workflowOID=(survey workflow OID)

Another call ?

This node prompts the caller if he/she wants to handle another call.

SetPropertyActivityImpl
INTERPRETER.message=Request another call ?

SetPropertyFromTextFileActivityImpl
URLspecifier=$$INSTANCE.website/doc/examples/CallCenter/templates/yesno_template.html
targetPropertyName=unformattedText

SetPropertyActivityImpl
formattedText="$$unformattedText"
SERVLET.RESPONSE.print=$formattedText
close=$SERVLET.RESPONSE.close

WaitActivityImpl …
SetPropertyActivityImpl
TARGET=$INSTANCE.answer

Wait

This node waits 2000 milliseconds before polling for a customer response from the server to the current request.

WaitActivityImpl
waitTime=2000

Interrupted ?

This node checks the Process Interrupt table to see if the caller decided to logout.

DBConnectActivityImpl
…

DBExecuteQueryActivityImpl
executeString= SELECT processID FROM ProcessInterrupts WHERE keyID=$INTERPRETER.callerID
…

DBResultSetActivityImpl
propertyNames=processID
…

DBCloseActivityImpl
…

Clear Interrupt & Set Active Call return to ERR

This node DELETES the client interrupt from the database and sets the call record return status to 'ERR'.

DBConnectActivityImpl …
DBExecuteActivityImpl
executeString= DELETE FROM ProcessInterrupts WHERE keyID=$INTERPRETER.callerID
…

DBExecuteActivityImpl
executeString="UPDATE SurveyCall SET resultCode='ERR' WHERE callID=$INTERPRETER.callID"
…

DBCloseActivityImpl …

Display Client Interrupted

This node informs the caller that the request for interrupt has been received. Note that no wait is used for this "ending" information.

SetPropertyActivityImpl
INTERPRETER.message=The Survey Client has been interrupted.

SetPropertyFromTextFileActivityImpl
URLspecifier=$$INSTANCE.website/doc/examples/CallCenter/templates/info_end_template.html
targetPropertyName=unformattedText

SetPropertyActivityImpl
formattedText="$$unformattedText"
SERVLET.RESPONSE.print=$formattedText

Display Logout

This node informs the caller that the request to exit the Call Center Client has been confirmed. Note that no wait is used for this "ending" information.

SetPropertyActivityImpl
INTERPRETER.message=Logout -- Thank you for using the Survey Client.

SetPropertyFromTextFileActivityImpl
URLspecifier=$$INSTANCE.website/doc/examples/CallCenter templates/info_end_template.html
targetPropertyName=unformattedText

SetPropertyActivityImpl
formattedText="$$unformattedText"
SERVLET.RESPONSE.print=$formattedText

Call Center Server Workflow

The Call Center Server workflow is responsible for detecting and handling requests from the survey for customers to be surveyed. The following is an "overview" diagram of the workflow.

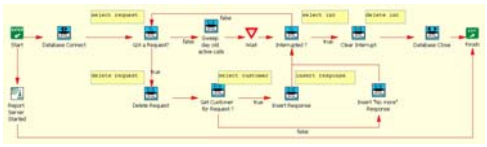

Figure 5 – Call Center Server Workflow

For a better understanding, the workflow has been broken into two parts, with a listing of its constituent workflow nodes, activities and properties.

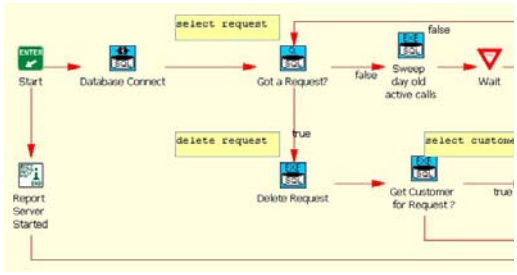

Figure 6 – Call Center Server Workflow (front)

Report Server Started

This node provides feedback to the server starter that the server has been started and continues to run after this page is displayed. To accommodate this "fork and continue" workflow, special care is taken after the page is submitted to the browser via SERVLET.RESPONSE.print to explicitly close the request. If the close was omitted the page will not be displayed.

SetPropertyActivityImpl
INTERPRETER.message=The Survey Server has been started.

SetPropertyFromTextFileActivityImpl
URLspecifier=$$INSTANCE.website/doc/examples/CallCenter/templates/admin_info_template.html
targetPropertyName=unformattedText

SetPropertyActivityImpl
SERVLET.RESPONSE.print="$$unformattedText"
close=$SERVLET.RESPONSE.close

Database Connect

This node connects to the database. This connection is shared by all the statements executed in the Call Center Server.

DBConnectActivityImpl
…

Got a Request?

This node checks to see if there are any requests from Call Center Clients in that database. Note that this node's Enter Activity explicitly sets the TOPOLOGY_STATE to "2" (BRANCH) due to the existence of a feedback loop after a fork. If this is not done, the workflow would treat it as a merge of the fork and wait.

EnterActivityImpl
TOPOLOGY_STATE=2

DBExecuteQueryActivityImpl
executeString=SELECT requestID,callerID,projectID FROM CallRequest
…

DBResultSetActivityImpl
propertyNames=INTERPRETER.requestID,INTERPRETER.callerID,INTERPRETER.projectID
returnStatusProperty=TARGET
…

Delete Request

This node DELETES requests from the database as they are handled by the Call Center Server.

DBExecuteActivityImpl
executeString=DELETE FROM CallRequest WHERE requestID=$INTERPRETER.requestID
…

Sweep day old active calls

This node checks to see if any calls have been unresolved (return code = NULL) for more that a day and DELETES the call. This will allow the corresponding customer to be surveyed.

DBExecuteActivityImpl
executeString=DELETE FROM SurveyCall WHERE resultCode IS NULL AND Day(CURRENT_TIMESTAMP) - Day(callTimeStamp) >= 1
…

Wait

This node waits 10000 milliseconds before polling for additional requests for customers to survey.

WaitActivityImpl
waitTime=10000

Get Customer for Request ?

This node represents the "heart" of the Call Center Server. It checks the database for unhandled customers for a given survey project. Its selection criteria to broker a customer to a caller is: any customer who hasn't been called, or has been called and has a status of 'ABN' (Aborted Call), 'BSY'

(Busy Number), ‘CBL’ (Call Back Later), ‘CAT’ (Call At Time) or ‘NAN’ (No Answer).

DBExecuteQueryActivityImpl
executeString=SELECT targetID FROM SurveyProject
WHERE projectID=$INTERPRETER.projectID
…

DBResultSetActivityImpl
propertyNames=INTERPRETER.targetID
…

DBExecuteQueryActivityImpl
executeString="SELECT sc.customerID FROM
SurveyCustomer AS sc WHERE
sc.targetID=$INTERPRETER.targetID AND
(SELECT COUNT(callID) FROM SurveyCall
WHERE customerId=sc.customerID AND
projectID=$INTERPRETER.projectID AND
targetID=$INTERPRETER.targetID) = 0 OR
(SELECT COUNT(callID) FROM SurveyCall
WHERE customerId=sc.customerID AND
projectID=$INTERPRETER.projectID AND
targetID=$INTERPRETER.targetID AND
callTimeStamp = (SELECT MAX(callTimeStamp)
FROM SurveyCall WHERE
customerId=sc.customerID AND
projectID=$INTERPRETER.projectID AND
targetID=$INTERPRETER.targetID) AND
(Day(CURRENT_TIMESTAMP) -
Day(callTimeStamp) >= 1 AND (resultCode = 'ABN'
OR resultCode = 'BSY' OR resultCode = 'CBL' OR
resultCode = 'CAT' OR resultCode = 'NAN'))) > 0”
…

DBResultSetActivityImpl
propertyNames=INTERPRETER.customerID
…

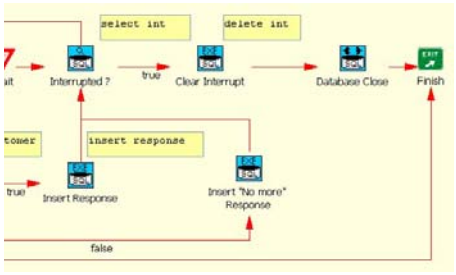

Figure 7 – Call Center Server Workflow (back)

Interrupted ?

This node checks the database to see if a request has been made to stop the server (keyID = “0”).

DBExecuteQueryActivityImpl
executeString= SELECT processID FROM
ProcessInterrupts WHERE keyID=0
…

DBResultSetActivityImpl
propertyNames=processID
…

Insert Response

This node INSERTS a record in the database in response to a caller’s request for a customer to survey.

DBExecuteActivityImpl
executeString=INSERT INTO CallResponse (callerID,customerID)
VALUES ($INTERPRETER.callerID,
$INTERPRETER.customerID)
…

Insert "No more" Response

This node INSERTS a response with a customer ID of 0 in answer to a caller’s request for a customer to survey. The 0 value signifies that there are no more customers to survey at the current time for the given survey project.

DBExecuteActivityImpl
executeString=INSERT INTO CallResponse (callerID,customerID)
VALUES ($INTERPRETER.callerID,0)
…

Clear Interrupt

This node DELETES the process interrupt for the server before proceeding to exit.

DBExecuteActivityImpl
executeString=DELETE FROM ProcessInterrupts WHERE
keyID=0
…

Database Close

DBCloseActivityImpl
connectionProperty=$INTERPRETER.DB_CONNECTION

This node closes the server-wide database connection.

Survey Workflow

The Survey workflow represents the survey presented to the customer. It is intended to direct the caller when he/she is on the phone with the customer. The following is an “overview” diagram of the workflow.

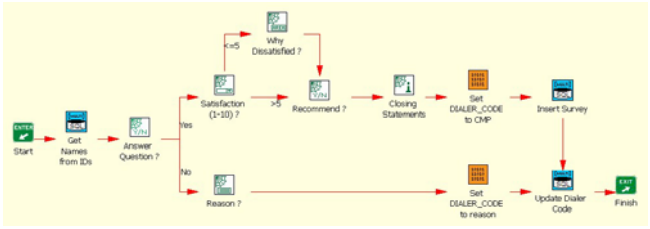

Figure 8 – Survey Workflow

For a better understanding, the workflow has been broken into two parts, with a listing of its constituent workflow nodes, activities and properties.

Figure 9 – Survey Workflow (front)

Get Names from IDs

This node SELECTS attributes from the database for callers, customers and surveys from the database and sets corresponding INTERPRETER properties.

SetPropertyActivityImpl
  INTERPRETER.surveyCompanyName=SurveyTakers
  …

DBExecuteQueryActivityImpl
  executeString=SELECT name from SurveyCaller
    WHERE callerID=$INTERPRETER.callerID
  …

DBResultSetActivityImpl
  propertyNames=INTERPRETER.callerName
  …

DBExecuteQueryActivityImpl
  executeString=SELECT name,phone from
    SurveyCustomer WHERE
    customerID=$INTERPRETER.customerID
  …

DBResultSetActivityImpl
  propertyNames=INTERPRETER.customerName,INTE
    RPRETER.customerPhone
  …

DBExecuteQueryActivityImpl
  executeString=SELECT name,businessType FROM
    SurveyTarget WHERE
    targetID=$INTERPRETER.targetID
  …

DBResultSetActivityImpl
  propertyNames=INTERPRETER.targetName,INTERP
    RETER.targetBusinessType
  …

DBCloseActivityImpl
  …

Answer Question ?

This node asks the customer if they would like to participate in the survey. The message used in this inquiry (and the others in this workflow) are specified in the servlet.config file.

SetPropertyActivityImpl
  INTERPRETER.message=$$INSTANCE.answerNodeQuestion

SetPropertyFromTextFileActivityImpl
  URLspecifier=$$INSTANCE.website/doc/examples/CallCenter/
    templates/yesno_template.html
  targetPropertyName=unformattedText

SetPropertyActivityImpl
  formattedText="$$unformattedText"
  SERVLET.RESPONSE.print=$formattedText
  close=$SERVLET.RESPONSE.close

WaitActivityImpl …
SetPropertyActivityImpl
  TARGET=$INSTANCE.answer

Note the activity pattern of (1) setting the properties to be used to generate/substitute into the page, (2) formatting a text template file that will constitute the contents of the page, (3) setting of the "SERVLET.RESPONSE.print" property to send the page to the browser, (4) waiting until the user hits a button on this page, and (5) getting the answer from this page to set the node's TARGET property (and subsequent path). This pattern is used by nodes that interact with a servlet to sequence browser pages.

Reason ?

This node asks the caller for the reason the customer chose not to participate in the survey. The possible reasons are presented as a list of options that are subsequently stored in the result code of the call.

SetPropertyActivityImpl
  INTERPRETER.displayList=Abandoned Call, Answering Machine,
    Busy Signal, Call Back Later, Call At Time, Deceased/Moved
    Customer, Dead Line, Duplicate Call, FAX Response,Language
    Problem, No Answer, Not Done Business, No Reason, Refused
    to Participate, Telco Error, Wrong Number
  INTERPRETER.valueList=ABN, AMC, BSY, CBL, CBT, DEC,
    DED, DUP, FAX, LNG, NAN, NDB, NUL, REF, TCO, WRG
  INTERPRETER.message="$$INSTANCE.reasonQuestion "

SetPropertyActivityImpl
  INTERPRETER.resultsRow='<option
    value="$CONTEXT.value">$CONTEXT.display</option>

SubWorkflowActivityImpl
  workflowOIDType=com.nouveausystems.alliance.service.object.OI
    DImpl
  executionWorkspaceID=NULL

SetPropertyActivityImpl
  INTERPRETER.options=$INTERPRETER.results

SetPropertyFromTextFileActivityImpl
  URLspecifier=$$INSTANCE.website/doc/examples/CallCenter/
    templates/choice_template.html
  targetPropertyName=unformattedText

SetPropertyActivityImpl
  formattedText="$$unformattedText"
  SERVLET.RESPONSE.print=$formattedText
  close=$SERVLET.RESPONSE.close

WaitActivityImpl …

SetPropertyActivityImpl
TARGET=$INSTANCE.option
INTERPRETER.reason=$INSTANCE.option

Satisfaction (1-10) ?

This node asks the customer for a satisfaction level (low to high) from 1 – 10. Note the use of the Compare Exit activity to choose the outgoing link.

SetPropertyActivityImpl
INTERPRETER.displayList=1,2,3,4,5,6,7,8,9,10
INTERPRETER.valueList=$INTERPRETER.displayList
INTERPRETER.message=$$INSTANCE.satisfactionNodeQuestion

SetPropertyActivityImpl
INTERPRETER.resultsRow='<option value="$CONTEXT.value">$CONTEXT.display</option>'

SubWorkflowActivityImpl
workflowOIDType=com.nouveausystems.alliance.service.object.OIDImpl
executionWorkspaceID=NULL

SetPropertyActivityImpl
INTERPRETER.options=$INTERPRETER.results

SetPropertyFromTextFileActivityImpl
URLspecifier=$$INSTANCE.website/doc/examples/CallCenter/templates/choice_template.html
targetPropertyName=unformattedText

SetPropertyActivityImpl
formattedText="$$unformattedText"
SERVLET.RESPONSE.print=$formattedText
close=$SERVLET.RESPONSE.close

WaitActivityImpl …
SetPropertyActivityImpl
INTERPRETER.satisfaction=$INSTANCE.option

CompareExitImpl
value1=$INTERPRETER.satisfaction
value2=5
comparison=GREATER THAN
trueLabel=>5
falseLabel=<=5

Why Dissatisfied ?

If the satisfaction level is 5 or less, this node asks for an elaboration via a multi-line text area.

SetPropertyActivityImpl
INTERPRETER.message=$$INSTANCE.dissatisfactionQuestion

SetPropertyFromTextFileActivityImpl
URLspecifier=$$INSTANCE.website/doc/examples/templates/CallCenter/textarea_template.html
targetPropertyName=unformattedText

SetPropertyActivityImpl
formattedText="$$unformattedText"
SERVLET.RESPONSE.print=$formattedText
close=$SERVLET.RESPONSE.close

WaitActivityImpl …
SetPropertyActivityImpl
INTERPRETER.dissatisfaction=$INSTANCE.text

Recommend ?

This node asks the customer for a Yes/No response as to whether they would recommend this target vendor to others. The Yes/No answer is then converted to a 1/0 value for storage in the database.

SetPropertyActivityImpl
INTERPRETER.message="$$INSTANCE.loyaltyNodeQuestion"

SetPropertyFromTextFileActivityImpl
URLspecifier=$$INSTANCE.website/doc/examples/CallCenter/templates/yesno_template.html
targetPropertyName=unformattedText

SetPropertyActivityImpl
formattedText="$$unformattedText"
SERVLET.RESPONSE.print=$formattedText
close=$SERVLET.RESPONSE.close

WaitActivityImpl …
SetPropertyActivityImpl
TARGET=$INSTANCE.answer

SetPropertyActivityImpl
INTERPRETER.loyalty=0

CompareActivityImpl
value1=No
value2=$INSTANCE.answer
comparison=0
skipCount=1

SetPropertyActivityImpl
INTERPRETER.loyalty=1

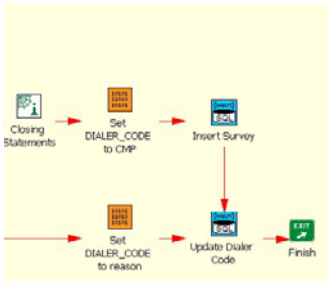

Figure 10 – Survey Workflow (back)

Closing Statements

This node prompts the caller to present a wrap-up to the customer upon completion of the survey.

SetPropertyActivityImpl
INTERPRETER.message=$$INSTANCE.closeNodeQuestion

SetPropertyFromTextFileActivityImpl
URLspecifier=$$INSTANCE.website/doc/examples/CallCenter/templates/info_template.html
targetPropertyName=unformattedText

SetPropertyActivityImpl
formattedText="$$unformattedText"
SERVLET.RESPONSE.print=$formattedText
close=$SERVLET.RESPONSE.close

```
WaitActivityImpl
   ...
```

Set DIALER_CODE to CMP

This node sets the INTERPRETER property for the result code to Complete ('CMP').

```
SetPropertyActivityImpl
   INTERPRETER.resultCode=CMP
```

Set DIALER_CODE to reason

This node sets the INTERPRETER property for the result code to the reason specified in the "Reason ?" node.

```
SetPropertyActivityImpl
   INTERPRETER.resultCode=$INSTANCE.reason
```

Update Dialer Code

This node sets the result code in the SurveyCall record in the database.

```
DBConnectActivityImpl
   ...
DBExecuteActivityImpl
   executeString="UPDATE SurveyCall SET resultCode =
      '$INTERPRETER.resultCode' WHERE callID =
      $INTERPRETER.callID"
   ...
DBCloseActivityImpl
   ...
```

Insert Survey

This node INSERTS a record representing the completed survey in the database.

```
DBConnectActivityImpl
   ...
DBExecuteActivityImpl
   executeString="INSERT INTO CompletedSurvey
      (callID,satisfaction,loyalty,dissatisfaction) VALUES
      ($INTERPRETER.callID,$INTERPRETER.satisfacti
      on,$INTERPRETER.loyalty,'$INTERPRETER.dissat
      isfaction')"
   ...
DBCloseActivityImpl
   ...
```

Installation

The Nouveau Alliance Call Center framework can be located in three locations:

- A pre-installed workspace called "examples".
- A workflow **template** called "examples".
- A ZIP archive file named CallCenter.zip that is available from the Nouveau Systems website.

The specific installation instructions depend on the location.

Pre-installed Workspace

If the "examples" workspace has already been installed, with workflow definitions (CallCenterClient, CallCenterServer, CallCenterStopProcess, WebSurvey and LoadDatabase), no additional installation is required. If these workflow definitions do not exist, please follow the ZIP Archive instructions presented below.

Workspace Template

To install the "examples" workspace from a template, start FlowSpace and select the New Workspace option from the File menu and select the "examples" template. Once the workspace is created, confirm that the Call Center workflows are present (see list above). If not, then follow the ZIP Archive instructions presented below.

ZIP Archive

The Call Center ZIP archive file can be downloaded from here:

http://download.nouveausystems.com/whitepapers/CallCenter.zip

To install this zip file:

1. Change directory to your Alliance root directory
2. Unzip the ZIP archive file in this directory. The output of the ZIP file will be placed in the following directories:

   doc/examples/CallCenter
   etc/servlets

3. Start FlowSpace and open or create a workspace called "examples".

4. Into this workspace, import the workflow definition archive files in this Call Center xml directory:

   CallCenterClient.zip
   CallCenterServer.zip
   CallCenterStopProcess.zip
   LoadDatabase.zip

   These XML-archive files specify workflow definitions that contain all of the Call Center operations.

   To load these definitions into the "examples" workspace, select the "Import New Workflow" option from the FlowSpace section page number pop-up menu in the lower-right corner of the display. You can select more that one xml archive file to import at a time by using the control or shift keys while making the selection.

Required License and Version

A Professional or Department license for FlowSpace version 3.2 and later is required to use the Call Center. If you're currently using Personal Edition, try FlowSpace Professional Edition *free* for 30 days now! Just point you web browser at your Nouveau Alliance server's home page, and click "Get Your Upgrade Evaluation!" to get started.

With the appropriate license, and a running Nouveau Alliance server, you can demonstrate the use of the Call Center.

Demonstration

To demonstrate the Call Center:

1. Make sure that the Nouveau Alliance Server is running.
   The workflows and the servlets of the framework require this server process.

2. Display the Call Center's administration page:

   http://yourhost:8081/doc/examples/CallCenter/admin.html

   This page contains buttons that control the Call Center's operation (loading data, starting and stopping processes).

3. Load in the demonstration database schema by pressing the Database Initialize button.
   This step loads in test survey data (survey projects, target vendors, the vendor's customers, and survey callers).

4. Start the Call Center Server by pressing the Call Center Server Start button.
   This step starts the server's workflow and runs until it is stopped by you. When you need to stop this server, press the Server Stop button.

5. Start a Call Center Client by pressing the Call Center Client Start button.
   Each client is invoked via a Nouveau Alliance Servlet and displays its interface via a web browser.

   This step will display a login page in a web browser.

6. Login to the Call Center Client
   The login page allows you to select a caller and survey project. Upon the submission of this information, a survey will be started.

Customization

You can replace the customer survey described in this white paper by specifying what data that you want to retain (the Database Schema) and how you want to obtain the data (the Survey Workflow).

Database Schema

You can populate or extend the database tables which store the callers, customers, and survey vendors and content.

Caller Data

The survey takers (callers) in the demonstration data are characterized by a "name" field in the SurveyCaller table. To build your own survey, you can populate this table with your callers and/or Alter the table to maintain more information about the caller (additional contact information).

Survey Project and Target Data

You need to specify your survey projects in the SurveyProject table and their survey target vendors in the SurveyTarget table. The SurveyProject table will need to be populated by entries specifying the title of the survey and the target vendor (by ID) to be surveyed. The SurveyTarget table will need to be populated by survey target vendors (including their name and business type). As will the SurveyCaller table, you may want to Alter the table to add contact information.

Customer Data

Once that you have determined the survey target vendors, you will need to populate the SurveyCustomer table with the customers each survey target. The demonstration contains the basic information about the survey customer (name, phone number and target vendor). You could Alter the table to include any special information about the customer such as purchased product(s) or special calling instructions.

Survey Question Data

The demonstration survey was a customer satisfaction survey, so its completed survey contained metrics related to determining how happy the customer is with the target vendor. In conjunction with the Survey workflow, you will need to Alter the CompletedSurvey table to store information related to your survey.

Workflow Definitions

Once you establish the vendor and customers to be surveyed and what information is to be obtained, you can then specify how the data is to be gathered. This is done by replacing the Survey Workflow.

Survey Workflow

You will need to replace the Survey workflow with your own question sequencing and dependencies. As stated previously, the survey database schema must be synchronized with flow of the workflow.

You can complete the customization by deciding what properties should by specified in the servlet.config file, or specified "in-line" in the workflow node activities. For example, in the demonstration survey, for ease of customization, all of the survey questions are defined as INSTANCE properties.

Summary

This white paper describes the construction of a Call Center Framework using Nouveau Alliance. Some of the technologies used include:

- Integrated Derby/DB RDMS
- Workflow Enabled Servlets
- Flexible Workflow Modeling

The paper describes how to model the Call Center Framework from a process, database and workflow perspective. It also includes a demonstration survey to bring the framework to life and provide insight on how to customize this framework for your own use.