# Table of Contents

1 Introduction .................................................................................................................. 5
   1.1 Purpose .................................................................................................................. 5
   1.2 Audience ............................................................................................................... 5

2 Scenario Builder Overview .............................................................................................. 6

3 Scenario Blocks .............................................................................................................. 8
   3.1 Accept .................................................................................................................... 8
   3.2 Add to Calling List ............................................................................................... 8
   3.3 Answer .................................................................................................................. 9
   3.4 Attached Data ....................................................................................................... 9
   3.5 Collect Digits ....................................................................................................... 9
   3.6 Comment ............................................................................................................... 11
   3.7 Connect Call ....................................................................................................... 11
   3.8 Connect Chat ..................................................................................................... 13
   3.9 DB Execute ......................................................................................................... 13
   3.10 EMail ................................................................................................................ 15
   3.11 Exception Handler ............................................................................................ 15
   3.12 Exit .................................................................................................................... 16
   3.13 Fetch URL .......................................................................................................... 16
   3.14 Find Agent ........................................................................................................ 17
   3.15 Get Next Record .............................................................................................. 19
   3.16 Get Statistics .................................................................................................... 20
   3.17 Get User Configuration .................................................................................... 20
   3.18 Goto ................................................................................................................... 20
   3.19 If ......................................................................................................................... 21
   3.20 Log ...................................................................................................................... 21
   3.21 Menu .................................................................................................................. 21
   3.22 Play Prompt ....................................................................................................... 23
   3.23 Record ............................................................................................................... 23
   3.24 Request Callback ............................................................................................. 24
   3.25 Request Input .................................................................................................... 25
   3.26 Request Skill or Service .................................................................................. 26
   3.27 Salesforce.com Delete ..................................................................................... 26
   3.28 Salesforce.com Insert ....................................................................................... 27
   3.29 Salesforce.com Screenpop ................................................................................ 27
   3.30 Salesforce.com Search ..................................................................................... 28
   3.31 Salesforce.com Update ..................................................................................... 28
   3.32 Save Survey Response ..................................................................................... 29
   3.33 Self-Service Provided ...................................................................................... 29
   3.34 Send Message ................................................................................................. 29
   3.35 Set Agent State ............................................................................................... 30
   3.36 Set Disposition ............................................................................................... 30
   3.37 Set Priority ..................................................................................................... 31
   3.38 Set Prompt Language .................................................................................... 31
   3.39 Set Variable .................................................................................................... 31
3.40 Start Another Scenario ................................................................. 32
3.41 Wait .................................................................................................. 32
3.42 Web Screen Pop ............................................................................. 32

4 List of Variables .................................................................................. 33

5 Expressions ......................................................................................... 36
  5.1 String Expressions ........................................................................ 36
  5.2 Integer Expressions ...................................................................... 36
  5.3 Floating Point Expressions ............................................................ 36
  5.4 Variables ....................................................................................... 36
  5.5 Built-in Functions .......................................................................... 36

6 Voice Segment Types ......................................................................... 38

7 Scenario Example .............................................................................. 40
1 Introduction

1.1 Purpose

ServicePattern Scenario Builder Reference Guide describes the building blocks of the ServicePattern scenario language. The function of each scenario block, its parameters, and usage are explained. An example of a typical scenario is provided in the last section of the guide.

For information about scenario management in the context of contact center configuration, such as association of scenarios with interaction access points, see ServicePattern Contact Center Administrator Guide.

1.2 Audience

This guide is intended for professionals responsible for design, development and testing of interaction processing logic in your contact center.

Participants are expected to be familiar with general principles of computer programming and to have a solid understanding of contact center operations and resources that are involved in such operations, including agents and teams, services and skills, schedules, and access points.
2 Scenario Builder Overview

For every customer interaction that enters your contact center, the ServicePattern solution has to process that specific interaction to determine what to do with it, e.g., what prompts or announcements to apply, what resources to queue for, what music to play, or when to over-flow to alternate resources. The logic of such automated interaction processing is defined in scenarios. Execution of a scenario with respect to a specific interaction is triggered by a particular event such as arrival of a call at a specific access number or initiation of a chat session from a specific web page.

Scenarios are designed and edited in the application called Scenario Builder. This application is started from the Contact Center Administrator application. For more information, see section Scenarios of ServicePattern Contact Center Administrator Guide.

Scenario Builder incorporates a graphical user interface to visually connect a sequence of functional blocks, thus building your scenario. This is done through a flowchart format that represents the sequence of interaction processing steps in the scenario. Different types of blocks perform different functions, such as playing prompts, collecting digits, and looking for available agents.

Each block has its own configuration attributes, which appear in the edit pane when the block is added to the flowchart or selected within the flowchart. The attributes specify the function represented by the block. For example, the Play Prompt block has an attribute that specifies which prompt shall be played when this block is executed in a specific processing step of a specific scenario.

A scenario typically processes blocks sequentially; however, some blocks have multiple paths or options that the scenario can take depending upon a certain condition or an outcome of the block execution. These paths are called conditional exits. Conditional exits enable you to determine how the Scenario responds to certain conditions that may occur during the processing of an interaction, such as an agent not responding, or a customer input.
The subsequent sections of this document describe specific scenario blocks, their functions, attributes and usage.
3 Scenario Blocks

3.1 Accept

The Accept block accepts the call from the network. This either starts ring back tone or allows the system to play announcements before or instead of answering a call. This is commonly referred to as early media. Early media denotes the capability to play media (audio or telephony) before a two-way voice session has been established. For telephony provisioning, establishment of media in the backwards direction may be desirable so that tones and announcements can be played prior to providing answer supervision. Note that call billing typically does not start until the call is answered or until a predefined carrier time is surpassed (in some instances this might be as high as 120 seconds, however, typically closer to 6 seconds).

Important:
1. Using the Accept block in a Scenario disables the default behavior of answering a call automatically on scenario start.
2. Announcement playback enables voice communication only to the caller, not from the caller. Use of Answer block will be required to establish two-way communication after the call is accepted.
3. Announcement playback delay duration may be limited or not available at all, depending on the telecom carrier.

Settings
The Accept block has the following settings:

- **Provide progress indication (ringback) only** – Plays a ring back tone to the caller. Typically considered if the carrier response is fast enough that customers hear no ring back tone.
- **Establish one-way voice communication** – Provides ability to play announcements or prompts to the caller.

3.2 Add to Calling List

The Add to Calling List block adds a calling record to the specified calling list. This block can be used, for example, to automatically redial abandoned or accidentally disconnected calls, or to organize a follow-up campaigns based on messages left by customers calling outside of contact center normal hours of operation.

Settings
The Accept block has the following settings:

- **Title text** – The name of the block instance.
- **List name** – The name of the list to which the record shall be added. Mandatory. Note that the name must be added manually since the scenario may be designed before the list itself is created.
- **Phone number** – Phone number to be dialed. Mandatory. Could be the calling ID (ANI) provided for the original call or a number entered by the customer via an IVR application.
- **Additional data** – Additional data that shall be added to the given calling list record in the key-value format, e.g., customer’s first and last name.

### 3.3 Answer

The Answer block provides answer supervision to the carrier and answers the call. This opens two-way voice communication with the caller and starts the caller’s billing cycle. Typically follows the use of an Accept block at some period in a scenario.

**Important:** Using the Accept block in a scenario disables the default behavior of answering a call automatically on scenario start and requires the Answer block to provide answer supervision.

### 3.4 Attached Data

The Attached Data block adds or changes custom data associated with interactions processed by the given scenario. The attached data is sent to the Agent Desktop applications when interactions are distributed to agents and can be provided to any other integrated applications. For example, this block can be used to populate custom fields in Salesforce.com activity history records.

**Settings**
- **Title text** – The name of the block instance.
- **add a data item** – Defines new data that shall be added to the given interaction as a new key-value pair, or specifies a new value for an existing key value pair. To populate custom fields in Salesforce.com activity history records, specify custom field names as data item names with the desired values.

### 3.5 Collect Digits

The Collect Digits block prompts caller to input a string of digits using the phone keypad and collects the digits.

1. The scenario plays a prompt requesting digit input and waits for input.
2. If the user does not start the input (does not enter the first DTMF digit) before the **Timeout Before First Digit** expires, the block plays the *Short version of main prompt* (e.g., *We did not receive a valid entry, please try again…*) and awaits input again, up to the number of attempts configured in the Retries field. If the specified Retries count is exceeded, the block exits via the **No Input** condition.
3. The caller enters the digits using the phone keypad.
4. The input is complete when the caller either enters the specified **Max Number of Digits**, or enters the *Finish Input button* (usually #), or interrupts the input for the specified **Timeout Between Digits**.
5. The input can be started over by entering the specified **Clear Input** digit (usually *), if necessary.
6. Upon completion, the block saves the entered digits as a scenario variable with the specified name.

**Conditional Exits**

The Collect Digits block has conditional exits to determine the action of the Scenario when the following events occur:
- **No Input** - The Scenario did not receive any input from the caller after repeating the *Short version of main prompt* for the number of times set in the *Retries* field.
- **Aborted** - The caller aborted input by pressing the *Abort input digit* specified in the block’s parameters.

### Prompts

The block has the following prompts:

- **Main Prompt** (*Prompt to play*) - The prompt the voice application plays to instruct the caller to input data. Mandatory.
- **Short Prompt** (*Short version of main prompt*) - The prompt the voice application plays when the *Timeout Before First Digit is Dialed* expires. Optional, by default *Main Prompt* will play.

To specify the prompt, click *select* and choose the desired from the *Prompts list* window. To create a new prompt, click the *Add new* button. For more information about creating prompts, see section *Voice Segment Types*.

### Settings

The block has the following settings:

- **Title text** - The name of the block instance.
- **Max number of digits to expect** - The maximum number of digits to expect. Leave this field empty if the number of digits may vary.
- **Finish input button** - The digit the caller enters to indicate the input is complete. Most voice scenarios use the pound sign (#) as this indicator. If you specify a Finish Input Digit, notify the customer about it during the Main prompt. Leave this field empty and specify Max Number of Digits if the input is a fixed length. If you do not require a Finish Input Digit, the Scenario uses the Max Number of Digits and Timeout Between Digits fields to determine when the caller completes the input.
- **Name of the variable to store the result** - The name for the scenario variable in which to store the entered digits.
- **Retries** - The number of times this block attempts to execute before the scenario moves to the next building block. Enter 1, or leave this field blank if you only want the block to attempt to execute one time.
- **Clear input digit** - The key the caller presses to reset input to empty string. This is useful when callers must enter long numbers. If you specify a Clear Input Digit, tell the customer about it during the Main Prompt (e.g., *If you make a mistake, press the star key*).
- **Abort input digit** - The digit the caller enters to clear all previously entered digits and abort the input. If you specify an abort input digit, tell the customer about it during the Main Prompt. If the caller presses the abort digit, the block will not perform validation (even if it is specified) and will immediately exit.
- **Timeout Before First Digit is Dialed** - The number of seconds the Scenario waits for the caller to start entering input before playing the *Short version of main prompt*.
- **Timeout Between Digits** - The number of seconds the scenario will wait for the next digit before input is considered completed.
3.6 Comment
The Comment block allows you to enter internal comments related to this scenario.

Settings
- **Title text** – The name of the block instance.
- **Comment** – Free-form text comments.

3.7 Connect Call
The Connect Call block connects a call to the destination specified in the Destination variable. If the destination extension has an agent logged in, the system tracks the agent’s state according to the state of the call. The block handles the call transfers and conferences internally and only ends when the remote party disconnects or last agent disconnects.

Note, that if there was a Find Agent block prior to the Connect Call, then queue treatment started by Find Agent continues until the Connect Call destination answers a call. If a service announcement prompt (Whisper) is specified for the Connect Call, the ring back tone or music on hold is played to the caller while the service announcement is played to destination party. The caller will not hear the announcement.

Conditional Exits
The Connect Call block has conditional exits to determine the appropriate action of the Scenario when the following events occur:
- **No Answer** - The destination phone rings but no one answers within the *No Answer Timeout*. This also includes other types of call failures except busy.
- **Busy** - The destination phone is busy (SIP 486) or the call is rejected by an agent.
- **Target Disconnected** - The call was answered by the destination party and eventually normally terminated from the destination side. This could be used, for example, for customer surveys when the agent disconnects first and a survey flow starts from this conditional exit.
- **Transfer Failed** – Failure branch for failed agent transfers.

Prompts
The Connect Call block has the following prompts:
- **Custom Hold Music** - The prompt the scenario plays when the caller is on hold. The prompt is always played from the beginning. The prompt is optional; if present it will override the default Hold and queue music treatment set at the contact center level (see section *Audio Treatments of ServicePattern Contact Center Administrator Guide* for more information).
- **Notification Prompt** - The optional prompt the scenario plays when a supervisor begins monitoring. The prompt can be made audible to the agent and the remote party, according to block’s parameters and is a short ding by default.
- **Service Announcement** - The optional prompt the scenario plays to inform the agent to which service an incoming call pertains or to play a beep as notification.
• **Custom ringback** – The prompt that will be played back to the caller instead of a standard ringback tone. Note that if the *Keep playing hold music while ringing on agent* option is selected in the preceding Find Agent block, the *Custom ringback* prompt will not be played even if specified. Instead the queue music will be played up to the moment of answer.

• **Repeated answer-side prompt** – Used for transfers of service calls to other call centers to announce the call information when the transferred call is answered by the remote agent, e.g., *This a call is from [name], please press [confirm answer button] to connect*. When the remote agent presses the confirm answer button, a call with the original calling party is established.

**Settings**

This block has the following settings:

- **Title text** - The name of the block instance.
- **Default Destination** - The default phone number to which the call connects if variable *Destination* is empty.
- **Override Destination** - The phone number to which the call connects. If this field has a value, the scenario ignores the Destination variable. Use this field only if you want to override the Destination variable.
- **Mark all calls connected by this block as overflow calls** – if selected, all calls connected via this block will be marked for reporting purposes as calls made to overflow destinations.
- **Allow call forwarding for routed agent destinations** - Enables call forwarding for destinations which have a corresponding agent logged in. The Connect Call block verifies if the destination phone belongs to a logged-in agent. If this is the case, by default it will disable call forwarding for this particular call so it will not be forwarding settings will disregarded. You can force the block to observe the call forwarding settings by selecting this box. This can be used, for example, to connect unanswered service calls to agents’ voicemail instead of sending them back to the service queue. For more information about call forwarding, see sections *Softphones* and *Hardphones* of the *ServicePattern Contact Center Administrator Guide*.
- **Override Calling Party Name with** - Enables you to override the configured Caller ID in the outbound call.
- **No Answer Timeout** - The number of seconds the scenario waits for a destination to answer the call before executing the *No Answer* conditional exit. The default is 10 seconds.
- **Auto-answer call in** – The number of seconds the scenario waits before the call is auto-answered. If you do not use auto-answer, leave this filed empty. This function will works for agents who use softphones. Support of this function in hardphones depend on a particular hardphone model.
- **Escape button for customer to hang up the agent**– The button that the caller can press to stop conversation with this agent. Unlike when it is released, the scenario will continue and can further process the call.
- **Monitoring Notifications** - Determines whether the *Notification Prompt* will be played to either the agent or the caller (or both) when monitoring of this call is started. To advise the remote party about call monitoring, select the *Notify Caller* checkbox. To advise the agent, select the *Notify Agent* checkbox.
- **Record Call**– Indicates to the scenario that the call shall be recorded. The recording starts when the call is answered and stops when it is released.
- **number of plays** – The number of times the *Service Announcement* prompt, if used, will be played to the agent
- **stop announcement button** – The button that the agent can use to interrupt playback of the service announcement prompt.
• **Confirm answer button** – The button that the agents of remote contact centers shall press to pick up service calls forwarded to them from your contact center after hearing the *Repeated answer-side* prompt.

• **Drop connection if no answer in** – The number of minutes the scenario will wait for the remote contact center to pick up the call before executing the *No Answer* conditional exit for this call. Starts from the moment when the remote agent answers.

### 3.8 Connect Chat

The Connect Chat block connects a chat to the agent found by the preceding *Find Agent* block or to the specified chat scenario entry. If the agent is logged in, the system tracks the agent’s state according to the state of the interaction. The block handles the call transfers and conferences internally and only ends when the remote party disconnects or last agent disconnects.

#### Conditional Exits

The Connect Call block has conditional exits to determine the appropriate action of the Scenario when the following events occur:

- **No Answer** - The chat request is submitted to the agent, but the agent does not answer within the *No Answer Timeout*.
- **Target Disconnected** - The chat request was answered by the agent and eventually normally terminated from the agent side. This could be used, for example, for customer surveys when an agent disconnects first and a customer survey flow starts from this conditional exit.

#### Settings

This block has the following settings:

- **Title text** - The name of the block instance.
- **Default Destination** - The default chat scenario entry to which the chat will be connected if no agents were found by the preceding *Find Agent* block to serve this interaction.
- **Override Destination** - The chat scenario entry to which the chat will be connected is this parameter is specified. If this field has a value, the scenario ignores the agent found by the preceding Find Agent block.
- **Mark all interactions connected by this block as overflow calls** – if selected, all chats connected via this block will be marked for reporting purposes as chats made to overflow destinations
- **No Answer Timeout** - The number of seconds the scenario waits for a destination to answer the call before executing the *No Answer* conditional exit. The default is 10 seconds.
- **Auto-answer in** – The number of seconds the scenario waits before the chat request is auto-answered. If you do not use auto-answer, leave this field empty.
- **HTML code** – Defines the page that will be presented to the customer when agent accepts the chat request.

### 3.9 DB Execute

The DB Execute block provides a way for a scenario to execute SQL statements on a specified database.
Conditional Exits
The DB Execute block has conditional exits to determine the appropriate action of the Scenario when the following events occur:

- **Failed** – error occurred during SQL statement execution. No error details are provided.
- **No Data** – the SELECT statement successfully executed but did not return any records.

Settings
This block has the following settings:

- **Title text** - The name of the block instance.
- **DB Connection** – The desired database connection. See below for details.
- **SQL Statement** – the SQL statement to be executed. SQL statements may use scenario variables.

  For example: `SELECT id, name FROM customers WHERE phone='$(item.from)'`

- **Recordset name** – for SELECT statements the name of the retrieved recordset should be specified. This allows scenarios to have more than one recordset and to choose the recordset to iterate on.

  The columns of the first retrieved record (if any) are stored in the scenario variables `<recordset_name>.<column_name>` (for example, RS.id). To iterate through the recordset use the "DB Next Record" block.

  The number of records in the retrieved recordset is limited to 25.

  The database connection is selected from a list of connections. Click the Manage DB connections button. The pop-up window will display all database connections defined in this scenario. For each connection the following data should be defined:

- **Name** – The name of the database connection. This name is shown in the DB Execute block connection selection menu.
- **JDBC driver and connection string** – Specify the JDBC driver and connection string that will be used to access this database. Note that templates are provided for some widely used DBMS systems.
- **Database user name and password** – Specify database access credentials.

The database connection selector allows the following operations:

- Add a new connection (Add New button)
- Edit and save the selected connection (Save button)
- Delete the selected connection (Delete button)
- Select the connection to be used in the DB Execute block (Select button)
- Clear the connection selected for the DB Execute block (Select None button)
- Close the window without changing the DB Execute block connection (Close button)
3.10 **EMail**

The Email block sends an email, with attachment if so configured. It can be used together with the Record block to send recorded voice messages. For general email configuration, see *ServicePattern Contact Center Administrator Guide*, section *Email Settings*.

**Conditional Exits**

The Email block has conditional exits to determine the appropriate action of the Scenario when the following events occur:

- **Mail not sent** – The message cannot be sent (for example, if the SMTP server is down).

**Settings**

The Email block has the following settings:

- **Title text** - The name of the block instance.
- **From / Display name** - The display name of the email sender. A scenario variable can be specified as the value using the $(varname) format.
- **From / Address** - The email address of the email sender. A scenario variable can be specified as the value using the $(varname) format.
- **To / Address(es)** - The email addresses of the intended recipients. If sending to multiple email addresses, separate email addresses using comma or semicolon. Scenario variables can be specified as values using the $(varname) format.
- **Message / Subject** - The subject line of the email. Can contain scenario variables in the $(varname) format.
- **Message / Format** - Specify the format for email: HTML (default) or Plain text.
- **Message / Body** - The text to be sent as a message body. Can contain scenario variables in the $(varname) format. ServicePattern does not impose any limits on the size of the email.
- **URL of the recording to attach** - The full HTTP URL of the voice recording previously done by a Record block. A scenario variable can be specified as the value using the $(varname) format. Typically, the Record block will store resulting URL in a scenario variable; this variable will be used by the Email block.

3.11 **Exception Handler**

The Exception Handler block provides an alternative branch the scenario can execute if an exception, a block error, or a disconnect occurs. This allows the scenario to continue executing instead of terminating as it normally would under such circumstances without the Exception Handler block. Use this block in any part of a scenario in which you expect exceptions, block errors, or caller disconnects to ensure continued processing.

The Exception Handler block has two conditional exits: **Try** and **Catch**.

- In the **Try** conditional exit, enter the sequence of blocks that you predict might generate an exception, block error, or disconnect.
- In the **Catch** conditional exit, enter the sequence of blocks that you want the Scenario to execute if an exception, block error, or disconnect occurs during the **Try** conditional exit.
The block initially executes the **Try** conditional exit.

- If an exception, block error, or caller disconnect occurs while executing any block in the **Try** conditional exit, the scenario executes the **Catch** conditional exit. After executing the **Catch** conditional exit, the scenario executes the next block in the flowchart.
- If no exception, block error, or disconnect occurs, the Scenario does not execute the **Catch** conditional exit, and instead processes the next block in the flowchart.

After execution, the Exception Handler block stores one of the following values in the Exception interaction property:

- **Disconnect** - The caller disconnected.
- **Error** - A block error or exception occurred.
- **No** - No exception occurred (normal processing).

### 3.12 Exit

The Exit block exits the scenario and disconnects the currently active interaction.

### 3.13 Fetch URL

The Fetch URL block fetches web content from a specified URL, using either the GET or POST requests. The received content may either be either be string or associative array (JSON, i.e. `{"varname1": "value1", "varname2": "value2"}`). In case of associative array, the values are stored in variables `<recordset_name>.[varname]`, for example, RS.varname.

**Conditional Exits**

The Fetch URL block has conditional exits to determine the appropriate action of the Scenario when the following events occur:

- **Failed** – error occurred during GET or POST method execution. No error details are provided.

**Settings**

The Fetch URL block has the following settings:

- **Title text** - The name of the instance of the block.
- **URL to Fetch** – URL text field
- **Username** - text field, variable substitutions allowed
- **Password** - text field, variable substitutions allowed
- **URL parameters** – click **add** to define URL parameters; variables in the `$(varname)` format can be used
- **Request type** – choose the request type, **GET** (default) or **POST**
- **Body** – data to be submitted, displayed only if POST is selected, variable substitutions allowed
- **Result is stored in** – choose where the received content will be stored. Select **Variable** if the entire response body is to be stored in the specified scenario variable; the body is not parsed. Select **Recordset** if a two-dimensional JSON encoded array is expected in response body; the array will be parsed and stored it in the specified recordset (provide a recordset name below). Use the **Get Next Record** block can be used to get records from the recordset.
The array should be in JSON format i.e.
{"record1":{"varname1":"value1","varname2":"value2"},"record2":{"varname1": "value3", "varname2": "value4"}}

The values are stored in variables <recordset_name>.<varname>, for example RS.varname1.
In all cases, the body of HTTP response is stored in $(fetchURLResultBody) variable and the operation result is $(fetchURLResultCode) variable.

The possible values of the $(fetchURLResultCode) variable are:
- 0: for successful response (response returned 200 OK)
- 1: 200 OK response, but unable to parse the body into recordset
- 2: 200 OK response, but body length exceeds 50Kb
- 3: Unable to connect to HTTP server or other connection errors
- Other: actual non-200 response code

3.14 Find Agent

The Find Agent block determines the most qualified agent to receive the interaction, and if/when the agent is available, creates a variable called Destination set to the agent’s phone number.

For example:
1. The scenario application collects data from a caller.
2. Based on the collected data, the scenario determines the qualifications necessary to handle the call.
3. The scenario finds an agent with the necessary qualifications, e.g., Peter at extension 151.
4. The scenario stores that agent’s extension internally in the Destination interaction property. In this example, the scenario would set the Destination interaction property to 151.
5. The scenario routes the call to the number stored in the Destination interaction property.

Important: Use the Connect Call block immediately after the Find Agent block to connect the call to the target agent.

Note that the block properties will be different depending on whether it is used in a voice or a chat scenario.

Conditional Exits

The Find Agent block has conditional exits to determine the action of the scenario when the following events occur:

- **No Agents** – This exit is taken if no agents with matching skills are logged in (or when the last such agent logs out before the call is routed.)
- **Queue Limit** – Your service provider may have set up a limit for the number of calls you can simultaneously have in a service queue. If the call processed by the given scenario exceeds this limit upon entering the queue, this exit will be used. Note that a repeated attempt to place the call in the same queue will result in termination of the scenario. Applies to voice scenarios only.
- **Escape Digit** – The caller presses the escape digit to exit the queue. This exit will be displayed only if the Escape button setting is defined (see below). Applies to voice scenarios only.
- **Callback** – This exit will be taken if the callback option (see below) is offered to the caller and is accepted by the caller. Applies to voice scenarios only.
- **Time Out** – This exit will appear only if you define one or more escalation intervals for *Agent skills required* (see below) provided that the last interval is finite. This exit will be taken if the last interval expires before any agents with matching skills become available. (Note that if the last matching agent logs out before the timeout expires, the *No Agents* exit will be taken.)

### Prompts

The Find Agent block used in voice scenarios has the following prompts:

- **Music on hold** - The prompt the scenario plays while the call is in queue. If not defined, the default *Music on hold and in queue* treatment will be played. The *Keep playing hold music while ringing on agent* parameter controls when the music is stopped; see below for details.
- **Initial Prompt** – An optional prompt that, if defined, will be played to the caller as soon as the call is places in queue (i.e., before the *Music on hold* starts).
- **EWT Announcement** - The prompt the voice scenario plays when providing the estimated wait time (EWT). The scenario uses the system to read the actual EWT. For example, the prompt announces *The estimated wait time is...*, and then the system announces the EWT, such as *eight minutes*.
- **Callback prompt** – The prompt that will be played to callers to give them an option of requesting a callback instead of waiting in the queue. See the description of the *Callback option* below for details.
- **Periodic reminder** - The prompt the scenario will periodically play to the call in queue at the frequency you set in the *Reminder Frequency field*. If you do not set this prompt, the reminder does not play. You can use the *EWT Announcement prompt* as a reminder prompt.

### Settings

The Find Agent block has the following settings:

- **Title text** - The name of the block instance.
- **Agent skills required** – A set of escalation intervals and required minimum skill levels for each. You may specify a number of escalation intervals; each interval may have different skill requirements. The last interval maybe finite or infinite (leave the end time empty to use an infinite interval). Use the *add skill* link to add skill requirements and specify the minimum skill level the agent must have in order to handle an interaction on each interval. Use 0 to exclude a skill on any particular interval.
- **Overflow call handling stats at** – defines at which escalation interval the found agent will be considered an overflow destination unless his skills/levels match the skill/level requirements of the preceding escalation intervals
- **Escape button** - The key on the telephone keypad (0-9, *, or #) that a caller can press to exit the queue. When the caller presses the escape digit, the scenario executes the *Escape Digit* conditional exit. Typically in such situations, the scenario sends the caller to a voicemail or terminates the call. Applies to voice scenarios only.
- **Keep call in queue** – If selected, the scenario will keep the call in queue even if there are no agents currently logged on. If not selected, the *No Agents* conditional exit will be used. Applies to voice scenarios only.
- **Callback option** – This option allows the callers to request a callback instead of waiting for an agent in the queue. The decision to offer the callback option is made based on the call’s estimated wait time (EWT) in queue. If a caller selects this option, his original inbound call will be
disconnected while the position of this call in the service queue will be preserved. The callback is made when it is the caller’s “turn” to be routed to an agent. Note that the callback option must be enabled at the service configuration level. For more information, see ServicePattern Contact Center Administrator Guide, section. To enable the option, select the enable if EWT is greater than checkbox and specify the minimum wait time that will enables the callback option. Specify the Callback button, i.e., the phone key that the caller will have to press to request a callback. If the callback is requested, the conditional exit Callback will be executed allowing the scenario to collect the callback data (see block Request Callback). Applies to voice scenarios only.

- **Periodic Reminder Frequency** - The number of minutes you want the scenario to wait between playing the Periodic Reminder prompt. Set this field only if you want the scenario to play the Periodic Reminder prompt. Enter 0 if you want to disable this feature. Applies to voice scenarios only.
- **Keep playing hold music while ringing on agent** – if selected, the Music on hold will continue after the Find Agent blocks exits; the prompt is stopped only when the subsequent Connect Call block actually connects the caller to the destination (destination answers). Otherwise, the caller will hear the ring-back tone from the moment his call is delivered to the agent and until the agent answers. This option only works if the block actually finds an agent; for all conditional exits the hold music stops immediately. Applies to voice scenarios only.
- **HTML code** – Defines the page that the customer will use during the chat session. Applies to chat scenarios only.

### 3.15 Get Next Record

The DB Next Record block provides a way for a Scenario to retrieve the next or previous record from a recordset created by a previously executed DB Execute or Fetch URL block.

#### Conditional Exits

The Get Next Record block has conditional exits to determine the appropriate action of the Scenario when the following events occur:

- **No more items** – No more items can be retrieved from the specified recordset.

#### Settings

This block has the following settings:

- **Title text** - The name of the block instance.
- **Direction** – Choose whether the next or the previous record shall be retrieved.
- **Recordset name** - The recordset that the record shall be retrieved from. Selected from the list of available recordsets. The list is populated from all DB Execute and Fetch URL blocks of the scenario.

The columns of the retrieved records (if any) are stored in the scenario variables RS.[name] (for example, RS.id).
3.16  Get Statistics
The Get Statistics block obtains one or more statistics and saves their values in scenario variables for future use.

Settings
The Get Statistics block has the following settings:

- **Title text** - The name of the block instance.
- **Variable Name** – Name of the variable where the statistic will be stored.
- **Statistic** – The type of statistic that is requested.
- **Service** – The service with respect to which the statistic is requested.

Scenario variables can be specified as values using the $(varname)$ format.

3.17  Get User Configuration
The Get user Configuration block obtains one or more of the properties of a user by his phone number, username, or email address and saves them in scenario variables for future use.

Settings
The Get User Configuration block has the following settings:

- **Title text** - The name of the block instance.
- **Find user by** – The type of the property that will be used to find the user.
- **Value** – The value of the property that will be used to find the user.
- **User properties to return** – Select the type of property that you want to use in the scenario and the name of the variable where its value will be stored.

Scenario variables can be specified as values using the $(varname)$ format.

3.18  Goto
The Goto block changes the order in which the scenario executes by redirecting the processing flow to a specified destination in the flowchart. To specify the destination of a Goto block:

1. Add the Goto block to the flowchart. The Scenario Builder highlights the block in red until you define the destination of the block.
2. Select the Goto block in the flowchart. The Edit pane displays a copy of the flowchart.
3. In the Edit pane, click the building block to which you want to redirect the flow using this Goto block.

The flowchart displays the new name of the Goto block, which indicates the location in the flowchart to which the block redirects the processing flow. The format of the name is: *Goto “[destination block title]”*. The Scenario Builder will highlight the Goto block in red if you remove its destination block during editing.
3.19 If

The If block allows branching of a scenario based on a flexible set of specified criteria. Multiple conditional exits (criteria) can be configured for each If block. Each criterion may consist of logical expressions that can include the following:

- Dialed number
- Caller’s number
- Current time
- Current date
- Day type
- Scenario variable (string)
- Scenario variable (number)
- Scenario variable (HOP)
- Estimated waiting time
- Current date and time

If a set of specified conditions for a criterion is satisfied, the scenario execution is continued from its conditional exit.

If the criteria’s conditions are not satisfied, the block following the If block is executed. The condition set may consist of multiple rows. Rows are joined by an AND (for the criteria to be satisfied, the condition in each column must be true) or OR (any one row must be true in order to satisfy the condition and return a true result).

Use the New Branch button until you have all the criteria and conditional exits necessary for this If block. With each set of criteria you create, additional tabs appear (labeled as Branch2, Branch3, etc.) and associated conditional exits appear in the flowchart.

3.20 Log

The Log block adds a message to the Scenario Engine log file. This block is intended for debugging and testing purposes only and may be removed for production versions of scenarios.

Settings

The Log block has the following settings:

- **Title text** - The name of the block instance.
- **Text message** - The free-form content of the message. You can use scenario variables in the text. For example: This is a test log message for call from $(item.from) to $(item.to).
- **Log Level** - Specifies the Scenario Engine log level where this message will appear.

3.21 Menu

The Menu block plays a menu prompt with options to choose from and then allows the caller to select an option by pressing a DTMF key.

1. The menu prompt announces options with corresponding keys from a phone keypad (0-9, *, #). Each selected option corresponds to a conditional exit.
2. The caller presses the key corresponding to the desired menu option.
3. The Scenario receives the input from the caller and processes the corresponding conditional exit.

If a key is not defined, it has no meaning and is considered invalid. The Menu block informs the caller if the input is invalid, or if the time allotted for the caller to enter digits expires. If a caller enters an invalid option or does not enter an option in the allotted time, the scenario executes the subsequent block.

To enable a key to become a valid option, select the checkbox corresponding to that key in the **Valid choices** section. When you enable a key, a text field appears in which you can enter a description of the key. The description you enter appears in the flowchart as the label of the conditional exit corresponding to the key. By default, the label is the key name.

**Conditional Exits**

You can configure a conditional exit for every telephone key (0-9, *, #). By default, the block has conditional exits for keys 1 and 2.

**Prompts**

The Menu block has the following prompts:

- **Main Prompt** (*Prompt to play*) - The prompt the scenario initially plays to the caller. Usually this prompt explains the available menu options. For example, the prompt can say *For customer service, press one, for technical support, press two, to speak with an operator, press the pound sign.*
- **Timeout Prompt** - The prompt the scenario plays when the time allotted for the caller to respond expires.
- **Invalid Prompt** - The prompt the scenario plays when the caller enters an invalid key.
- **Short Prompt** (*Short version of main prompt*) - The short version of the main prompt the scenario plays after the Timeout and Invalid Prompt. This prompt can remind the caller about the available menu options.

To specify the prompt, click **select** and choose the desired from the **Prompts list** window. To create a new prompt, click the **Add new** button. For more information about creating prompts, see section **Voice Segment Types**.

**Settings**

The Menu block has the following settings:

- **Allow interrupting prompt by a phone button** – Select to allow callers to interrupt the prompts by entering the desired option at any time; uncheck if input is not allowed until the prompt is complete.
- **Input Timeout** - The number of seconds the scenario waits for a caller’s input after playing the initial prompt or its short version before playing the Timeout prompt. If left blank, the block will wait for input indefinitely.
- **Retries** - The maximum number of times the voice application will allow the caller to provide input after the input timeout expiration or invalid input. If the number of retries is reached without successful input, the scenario moves to the next block in the flowchart. This parameter can either be set explicitly or via a scenario variable. This parameter can be set to zero to exit menu without retries.
3.22 Play Prompt

The Play Prompt block plays a voice or music prompt. This building block is also useful for reporting an error or the outcome of an operation for testing purposes. For scenarios in which input from the caller is expected in response to a prompt, use the Menu or Collect Digits blocks.

Prompts

The Menu block has the following prompts:

- **Prompt to play** - The prompt the scenario will play to the caller. Select (or create) the prompt that this block will play.

To specify the prompt, click **select** and choose the desired from the **Prompts list** window. To create a new prompt, click the **Add new** button. For more information about creating prompts, see section **Voice Segment Types**.

Settings

- **Allow interrupting prompt by phone button** – Select to allow callers to interrupt the prompt by pressing a phone key. If this checkbox is selected, and the caller inputs one or more digits, the system saves the digits and uses them in the block immediately following this Play Prompt block.
- **Next block will use this button as input** - Select this check box if the digits entered by the user should be preserved for potential use in the next block. If there is another Play Prompt block after this one, the next prompt will be interrupted immediately.

3.23 Record

The Record block provides a way for a scenario to record a message and optionally store it in the system database.

Initially the block records to a local voice file. When the recording is finished, the block may optionally play the recorded message back to user and offer one of three options: accept recording, record again, cancel recording.

If the recording is confirmed (or if confirmation is not required) the block can keep the recording in a file or upload it to the database and saves the recording ID and full HTTP URL in the specified scenario variables (see below).

Conditional Exits

The Record block has a conditional exit to determine the action of the scenario when the following event occurs:

- **Silence** - Executes when silence is detected. The **Detect Silence** check box must be selected and the **Cancel on initial silence of** parameter must be specified.
- **Error** - Executes when a block error occurs.
- **Max Recording Time Exceeded** - Executes when the **Maximum duration** is exceeded.
- **Cancelled** – Executes if the recording is cancelled by the user.
Prompts
The Record block has the following prompts:

- **Confirmation menu prompt** - The block plays this prompt after replaying the recorded message. This prompt will normally offer the following options: accept the recording, record again, and cancel recording. The user will be expected to select the desired option by pressing the phone key specified for each option (by default, these are digits 1, 2 and 3 respectively).

Settings
The Record block has the following settings:

- **Title text** - The name of the block instance of the block.
- **Existing recording ID** – The identifier of an existing database recording. The ID may be specified explicitly or as an application variable in form $(v\text{ar}name)$. If specified, the existing recording will be overwritten.
- **Maximum duration** - Specifies the maximum duration of the new portion of the recording in seconds.
- **Name of the variable to store resulting recording ID** - This parameter specifies the name of the scenario variable that will store the identifier of this recording.
- **Name of the application variable to store resulting recording URL** - This parameter specifies the name of the scenario variable that will store the full HTTP URL of the recording. This is useful when the caller needs to listen to the message that was just recorded.
- **Detect silence** - If selected, the block will analyze the voice stream in order to reject empty messages and determine the end of message. Note that this option requires more processing power.
- **Cancel on initial silence of** - The block will use the Silence exit if no voice energy is detected during the specified number of seconds.
- **Silence cut-off** - The block will stop recording after the specified number of seconds of continuous silence is detected following a period of voice energy.
- **Confirm before saving** – If selected, the block will play the recording back to the caller and ask the caller to confirm or re-record the message before it is saved into the database.
- **Beep at start of Recording** – If selected, the block will provide a tone indicating the beginning of the recording.
- **Store recording in the database** – If selected, the recording will be stored in the database. If not, it is stored in the local file only and can be used only within the same scenario.

3.24 Request Callback
The Request Callback block places a callback request in the service queue and waits for a matching agent. The block uses the skill requirements set in the last Find Agent block.

Conditional Exits
The Request Callback block has a conditional exit to determine the action of the scenario when the following event occurs:

- **No Agents** - Executes if no matching agents are logged in (or the last matching agent logs out while the callback request is waiting in the queue).
• **Time Out** - Executes if no matching agents are found or predicted to be found within the time specified in the *Cancel request after* parameter.

• **No Answer** - Executes if the party that requested the callback does not answer when called back. This exit is also used for any other reason the callback attempt fails except when the party is busy.

• **Busy** – Executes if the party that requested the callback is busy when called back.

**Prompts**

The Request Callback block has the following prompts:

• **Confirmation prompt** - The block plays this prompt to confirm that the callback request has been accepted. The caller may release the call at any time during the prompt. Otherwise the block will release the call as soon as the prompt is finished.

**Settings**

The Request Callback block has the following settings:

• **Title text** - The name of the block instance.

• **Callback phone** – The phone number to be called during the callback attempt. The caller ID (ANI) of the party requesting the callback is be used by default.

• **Cancel request after** - Specifies the time after which the request will be canceled if no matching agents can be found.

• **Start dialing** – The moment when the callback is to be made is determined by expiration of the estimated wait time (EWT). Slightly in advance of this moment, the system will dial the **Callback phone** automatically and connect a successful callback attempt to an agent that is supposed to become available by then according to the EWT prediction. This setting defines how many seconds in advance of the EWT expiration the callback attempt shall be made. Note that if an agent becomes available for this call before the EWT expiration, the system will dial customer as soon as that agent is **Ready**.

• **No Answer Timeout** – Specifies for how long the block will wait the party to answer before taking conditional exit *No Answer*.

### 3.25 Request Input

The Request Input block is used to request information from a chat customer. It can be used, for example, to ask for customer information at the beginning of the chat session (first and last name, email address, etc.), or to offer the customer a satisfaction survey at the end of the session.

**Settings**

The Request Input block has the following settings:

• **Title text** - The name of the block instance.

• **HTML code** – Defines the page that will be presented to the customer to request information.
3.26 Request Skill or Service

The Request Skill or Service block sets the skill requirements for the given interaction. Note that the minimum skill levels are set in the Find Agent block.

**Settings**

The Request Skill or Service block has two drop-down lists. The first one lists the skill groups currently configured in the system. The second one lists the skills belonging to the skill group selected in the first list.

1. In the first drop-down list, select the skill group containing the skill an agent must have to be considered qualified to handle this interaction.
2. In the second drop-down list, select the skill an agent must have to be considered qualified to handle the interaction.
3. The label of the Request Skill block will appear in the flowchart in this format: *Request Skill or Service* “ [skill group]=[skill]”

To remove all possible prior skill requirements for this interaction and to use only the skills specified in this block, select the **Reset prior skill requirements** checkbox.

3.27 Salesforce.com Delete

The Salesforce.com Delete block deletes the specified Salesforce.com (SFDC) object from the SFDC database.

**Conditional Exits**

The Salesforce.com Delete block has a conditional exit to determine the action of the scenario when the following event occurs:

- **Failed** – Executes if the delete operation failed.
- **No Data** - Executes in the specified object is not found.

**Settings**

The Salesforce.com Delete block has the following settings:

- **Title text** - The name of the block instance.
- **Object type name** – Type of the SFDC object to be deleted as defined in the SFDC application. May be specified as an application variable in form $(varname).
- **Object ID** – Identifier of the SFDC object to be deleted as defined in the SFDC application. May be specified as an application variable in form $(varname).
3.28 Salesforce.com Insert

The Salesforce.com Insert block creates the specified Salesforce.com (SFDC) object in the SFDC database. Note that to populate the custom fields in SFDC activity history records, the Attached Data block shall be used.

Conditional Exits

The Salesforce.com Insert block has a conditional exit to determine the action of the scenario when the following event occurs:

- Failed – Executes if the insert operation failed.

Settings

The Salesforce.com Insert block has the following settings:

- **Title text** - The name of the block instance.
- **Object type name** – Type of the SFDC object to be created as defined in the SFDC system. May be specified as an application variable in form $(varname).
- **Variable name of object ID** – Name of the variable that will be used as identifier for the SFDC object to be created. Will be set only if the block succeeds.
- **Object fields** – Object properties. Click *add* and specify the property *Name* as defined in the SFDC system and the desired *Value*. Repeat for the remaining object properties. Field values may be specified as application variables in form $(varname).

3.29 Salesforce.com Screenpop

The Salesforce.com Screenpop block specifies the Salesforce.com (SFDC) data to be displayed for the agent when the interaction is connected to this agent through the Connect Call block.

Settings

The Salesforce.com Screenpop block has the following settings:

- **Title text** - The name of the block instance.
- **Screenpop action** – Type of data to be displayed. Use the *Show object* option when the scenario can identify the object associated with the interaction. An SFDC page with the object properties will be displayed to the agent. The *Show query results* option can be used if such a record cannot be identified precisely, but the available interaction data can be used as search criteria for relevant records. An SFDC page with search results will be displayed.
- **Object ID** – Identifier of the SFDC object to be displayed. Must be specified if the *Show object* option is selected. May be specified as an application variable in form $(varname).
- **Search terms** – Search criteria. May be specified as application variables in form $(varname). Must be specified if the *Show query results* option is selected.
3.30 Salesforce.com Search

The Salesforce.com Search block executes the specified SOQL (Salesforce Object Query Language) statement. To use scenario variables in the statement, include them as $(varname). For example, `SELECT id, name FROM Accounts WHERE phone = '$(ANI)'.` The columns of the first record of retrieved recordset are stored in variables `<recordset_name>.<column_name>`. For example, for the statement above the results will be stored in variables `Recordset.id` and `Recordset.name`. To iterate through the recordset, use `Get Next Record` block. Note that only the first 25 records can be accessed.

Conditional Exits

The Salesforce.com Search block has a conditional exit to determine the action of the scenario when the following event occurs:

- **Failed** – Executes if the search operation failed.
- **No Data** - Executes if no data matching the specified search criteria is found.

Settings

The Salesforce.com Search block has the following settings:

- **Title text** - The name of the block instance.
- **Name the retrieved recordset** – Name of the recordset that will be retrieved via this search operation.
- **SOQL statement** – Record selection statement in the Salesforce Object Query Language. May contain application variables in form $(varname).

3.31 Salesforce.com Update

The Salesforce.com Update block updates properties of the specified Salesforce.com (SFDC) object. Note that to populate the custom fields in SFDC activity history records, the `Attached Data` block shall be used.

Conditional Exits

The Salesforce.com Update block has a conditional exit to determine the action of the scenario when the following event occurs:

- **Failed** – Executes if the update operation failed.
- **No Data** - Executes in the specified object is not found.

Settings

The Salesforce.com Update block has the following settings:

- **Title text** - The name of the block instance.
- **Object type name** – Type of the SFDC object to be created as defined in the SFDC system. May be specified as an application variable in form $(varname).
- **Object ID** – Identifier of the SFDC object to be updated. May be specified as an application variable in form $(varname).
• **Fields to update** – Object properties to be updated. Click *add* and specify the property *Name* as defined in the SFDC system and the desired new *Value*. If necessary, repeat for the other object properties to be updated. Field values may be specified as application variables in form $(varname)$.

3.32 **Save Survey Response**

The Save Survey Response block saves the customer survey response data for a completed interaction (three questions) for reporting purposes. Survey results are stored in scenario variables in the $(varname)$ format. The three survey questions are typically some variations of the following:

1. Was your issue resolved on your first contact with us?
2. On a scale from one to nine, how would you rate your overall satisfaction with our service?
3. On a scale from one to nine, how likely are you to recommend our product to your friend/colleague?

**Settings**

The Save Survey Response block has the following settings:

- **Issue Was Resolved** – The response is stored in the $(first_call)$ variable as required for reporting.
- **Contact Satisfaction** - The response is stored in the $(contact_satisfaction)$ variable as required for reporting.
- **Net Promoter Score Data** - The response is stored in the $(NPS_raw)$ variable as required for reporting.

3.33 **Self-Service Provided**

The Self-Service Provided block marks the call as successfully served by the IVR application. The Self-Service indicator is stored in the call detailed records and can be used in reports to distinguish the calls successfully served by an IVR from the calls that were simply abandoned by callers at that stage of processing.

3.34 **Send Message**

The Send Message block is used to send a message to a chat customer.

**Settings**

The Send Message block has the following settings:

- **Message** - The text of the message to be sent to the customer. Variables in the $(varname)$ format can be used in the message text.
3.35 Set Agent State

The Set Agent State block supports agent authentication via phone (i.e., using extension and PIN) when agents select the Dial-in and keep line open option for their Phone device. Also known as nailed connection, this option allows agents to establish a phone connection with the system from any phone and use this established physical connection for handling of subsequent calls within their login sessions.

A scenario that supports authentication via phone will usually begin with a Collect Digits block, which prompts the caller to enter their extension number and PIN and stores them in variables. The Set Agent State block then verifies the supplied extension number and PIN against the user configuration to authenticate the caller. If the agent is successfully authenticated, the block returns his current state and stores it in variable $(agentState). If the state is Not Ready, the Not Ready reason will be stored in variable $(notReadyReason). ServicePattern provides a scenario template to support the above process. The template is called External Agent Dial In.

Note that instead of reporting the agent’s current state, the block can be used to switch the agent to another desired state.

Conditional Exits

The Set Agent State block has a conditional exit to determine the action of the scenario when the following event occurs:

- **Authentication Failed** – The supplied extension number and PIN combination does not match credentials of any configured user.

Settings

The Set Agent State block has the following settings:

- **Agent’s digital ID (extension)** - Agent’s extension number or the name of the variable which holds the extension number, e.g., $(varPhone).
- **Over the phone agent PIN** – Agent’s PIN or the name of the variable which holds the agent’s PIN, e.g., $(varPIN).
- **Action** – The desired action; either get the current agent’s state or set it to one of the listed state. Note that for the agent authentication scenario described above the action must be set to GET_STATE.

3.36 Set Disposition

The Set Disposition block allows scenarios to specify a disposition for the interaction.

Settings

The Set Disposition block has the following setting:

- **Set Disposition** – The disposition name or the associated numeric code. If the disposition with the specified name or numeric code cannot be found, the interaction disposition will be set to Disposition Not Found.
3.37 Set Priority

The Set Priority block specifies the priority of the interaction relative to other interactions waiting in the service queue. The default interaction priority is 1.

Example:
Specify 2 to double the speed with which the given interaction will move through the queue relative to interactions with priority 1. Specify 0.5 to make the interaction move through the queue at half the speed of interactions with priority 1.

Settings
The Set Priority block has the following setting:

- **Priority** - Enter the value of the priority relative to other interactions in queue.

3.38 Set Prompt Language

The Set Prompt Language block determines the language in which the blocks following this block will play their prompts. This block is useful in the conditional exits of the Menu block that offers a choice of languages to a caller.

Note: Each application you develop has built-in multilingual prompt management. You can add or remove languages using the Prompt Manager. Each prompt has versions for all languages defined in the system. By default, English is always defined.

The Set Prompt Language block has a drop-down list containing all languages defined for your scenario. You can enable additional languages for the scenario in the Prompt List dialog box. To configure the Set Prompt Language block, select the language in the drop-down list in which you want the Scenario to play prompts, and click the Update button. The label of the Set Prompt Language block changes in the flowchart changes: *Set Prompt Language [language]*.

Once the Set Prompt Language block sets the language, each subsequent block plays prompts in that language until another Set Prompt Language block changes the language.

3.39 Set Variable

The Set Variable block sets a value for a scenario variable.

Settings
The Set Variable block has the following settings:

- **Variable name** - The name of the variable.
- **Value** – The desired variable value. Variables in the $(varname) format can be used as values. Values can be specified either as literal strings or expressions. Literal strings are passed exactly as entered. Expressions must begin with assignment sign = as the first character. For example, 2+2 will produce 2+2, whereas =2+2 will produce 4. Expression result produces one of the following data types: Strings, Integers and Floating point numbers. For more information, see section *Expressions*.
3.40 Start Another Scenario

The Start Another Scenario block starts another from within the given (parent) scenario. Variables defined in the parent scenario carry over to the sub scenario. After the sub scenario finishes executing, control returns to the parent scenario (except when terminated by error or disconnect). The parent scenario will resume by executing the next block in the flowchart.

To configure the Start Another Scenario block, select the desired sub scenario in the drop-down list. The label of the block in the flowchart changes to: Start Another Scenario ”[scenario name]”.

3.41 Wait

The Wait block will pause scenario execution for a specified number of seconds.

Settings

The Wait block has the following settings:

- **Wait duration** - The number of seconds to pause scenario for. Decimal fractions are allowed.

3.42 Web Screen Pop

The Web Screen Pop block specifies what to display for the agent when the interaction is connected to this agent through the Connect Call block.

Settings

The Web Screen Pop block has the following settings:

- **Title text** - The name of the block instance.
- **Open in popup window** – Indicates whether the pop-up should occur in the Context Information Area (right hand pane) of the Agent Desktop application (default) or in a separate window.
- **Screenpop action** – Indicates whether the pop-up will open a web page (default) or display text
- **URL of the page to open** – Specifies the URL of the web page that should be displayed if the Open web page screenpop action is selected. A query string can be added to supply variables for the screenpop page, e.g., http://www.localhost.com/Webform2.aspx?accountnumber=$(custNum)
- **Text to display** – Specifies the text to be displayed if the Display text screenpop action is selected.
# 4 List of Variables

The variables used in ServicePattern scenarios are described in the following table:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Local Variables</strong></td>
<td></td>
</tr>
<tr>
<td>$(destination)</td>
<td>Destination for the Connect Call or Connect Chat block</td>
</tr>
<tr>
<td>$(fetchURLResultBody)</td>
<td>The body of HTTP response received by most recent Fetch URL block</td>
</tr>
<tr>
<td>$(fetchURLResultCode)</td>
<td>The code of HTTP response received by most recent Fetch URL block</td>
</tr>
<tr>
<td>$(agentState)</td>
<td>The state of the agent obtained by most recent Set Agent State block</td>
</tr>
<tr>
<td>$(notReadyReason)</td>
<td>The Not Ready reason of the agent obtained by most recent Set Agent State block</td>
</tr>
<tr>
<td>$(doNotRecord)</td>
<td>Can be set to 1 or true (case insensitive) prior to Connect Call block to disable call recording. This overrides recording schedule or Record check box in Connect Call block.</td>
</tr>
<tr>
<td>$(XXX)</td>
<td>Local variables; not defined initially. Used to pass the data between scenario blocks. XXX is the name of the variable. E.g., $(accountNumber)</td>
</tr>
<tr>
<td><strong>Screenpop data</strong></td>
<td></td>
</tr>
<tr>
<td>$(screenpopData.XXX)</td>
<td>List of screenpop data received or set by IVR. Actual list of available screenpop data elements depends on the particular IVR and integration</td>
</tr>
<tr>
<td><strong>Interaction Properties</strong></td>
<td></td>
</tr>
<tr>
<td>$(item.media)</td>
<td>Media type. Can be set to voice or chat</td>
</tr>
<tr>
<td>$(item.from)</td>
<td>Origination address (phone number or chat user display name). Also known as ANI.</td>
</tr>
<tr>
<td>$(item.to)</td>
<td>Destination address (phone number or chat launch point name). Also known as DNIS</td>
</tr>
<tr>
<td>$(item.priorTo)</td>
<td>Prior destination address (phone number or chat launch point name); used for forwarded calls.</td>
</tr>
<tr>
<td>$(item.firstName)</td>
<td>Customer first name. Scenario may get or set this variable. Setting the variable also updates historical database interaction step as well as customer chat party information.</td>
</tr>
<tr>
<td>$(item.lastName)</td>
<td>Customer last name. Scenario may get or set this variable. Setting the variable also updates historical database interaction step as well as customer chat party information.</td>
</tr>
<tr>
<td>$(item.fullName)</td>
<td>Customer full name. Scenario may get or set this variable. For voice calls it is set to display name as received from the network. For chat this is initially empty, but can be set by scenario.</td>
</tr>
<tr>
<td>$(item.email)</td>
<td>Customer e-mail address. Scenario may get or set this variable.</td>
</tr>
<tr>
<td>$(item.sendTranscript)</td>
<td>For chat interactions, scenario must set this variable to 1 to have session transcript emailed to the customer when session ends. Can be set at any moment.</td>
</tr>
<tr>
<td>$(item.data.&lt;name&gt;)</td>
<td>Optional additional interaction data (key-value list). E.g., $(item.data.firstName)</td>
</tr>
<tr>
<td>$(item.message)</td>
<td>For chat interactions, last chat message received from the origination side.</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$(item.newMessage)</td>
<td>For chat interactions, adds new message as it was received from customer. Used to populate chat session with form field values. New message is treated as it was received from customer and is added to chat transcript.</td>
</tr>
<tr>
<td>$(item.transcript)</td>
<td>For chat interactions, full transcript of the chat session</td>
</tr>
<tr>
<td>$(item.externalChatData)</td>
<td>For chat interactions, full collection of data received from customer chat page</td>
</tr>
<tr>
<td>$(item.virtualDevice)</td>
<td>Voice scenario may set this variable to indicate that inbound call should not be terminated when scenario ends; instead call will be converted to &quot;virtual phone&quot; (used for external agents on PBX).</td>
</tr>
</tbody>
</table>

**Scenario Configuration Properties**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(app.version)</td>
<td>Scenario version (as reported by configuration server)</td>
</tr>
<tr>
<td>$(app.appName)</td>
<td>Scenario name</td>
</tr>
<tr>
<td>$(app.name)</td>
<td>Scenario entry name</td>
</tr>
<tr>
<td>$(app.priority)</td>
<td>Priority as specified in scenario entry</td>
</tr>
<tr>
<td>$(app.tenantName)</td>
<td>Tenant name</td>
</tr>
</tbody>
</table>

**Custom Parameters**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(app.custom.XXX)</td>
<td>Custom parameters which can be specified in scenario entry. Must follow naming convention $(app.custom.XXX) where XXX is name of the custom parameter. All scenario variables conforming to such convention are treated as custom parameters. The list of such parameters will be displayed in every scenario entry associated with the given scenario. You can set values of such custom parameters for every scenario entry point separately. For more information, see description of property Scenario Parameters in sections Dial-in Scenario Entries and Chat Scenario Entries of ServicePattern Contact Center Administrator Guide.</td>
</tr>
</tbody>
</table>

**User Configuration Properties**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(user.version)</td>
<td>Version of user's configuration information</td>
</tr>
<tr>
<td>$(user.loginId)</td>
<td>User's username.</td>
</tr>
<tr>
<td>$(user.team)</td>
<td>Name of the team the user is assigned to</td>
</tr>
<tr>
<td>$(user.firstName)</td>
<td>User's first name</td>
</tr>
<tr>
<td>$(user.lastName)</td>
<td>User’s last name</td>
</tr>
<tr>
<td>$(user.phone)</td>
<td>The phone number the user is logged on to or user's extension as specified in configuration</td>
</tr>
</tbody>
</table>

**Service Configuration Properties**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(service.name)</td>
<td>Name of the service (as specified in the scenario entry or scenario's Request Skill or Service block)</td>
</tr>
<tr>
<td>$(service.version)</td>
<td>Version of service configuration information</td>
</tr>
<tr>
<td>$(service.slPercent)</td>
<td>Service level call percentage</td>
</tr>
<tr>
<td>$(service.slTimeout)</td>
<td>Service level timeout in seconds</td>
</tr>
<tr>
<td>$(service.EWT)</td>
<td>Current estimated waiting time for the service</td>
</tr>
</tbody>
</table>

**Outbound Target Work Item Parameters**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(workitem.firstName)</td>
<td>Customer's first name</td>
</tr>
<tr>
<td>$(workitem.lastName)</td>
<td>Customer’s last name</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$(workitem.suggestedPhone)</td>
<td>Customer's default phone number</td>
</tr>
<tr>
<td>$(workitem.otherInfo)</td>
<td>All list fields except phones, first name and last name. E.g., $(workitem.otherInfo.MTN)</td>
</tr>
<tr>
<td>$(workitem.fullInfo)</td>
<td>All list fields. E.g., $(workitem.fullInfo.MTN)</td>
</tr>
</tbody>
</table>
5 Expressions

5.1 String Expressions

- Strings are enclosed in double quotes, e.g. “sample string”
- Backslash can be used to embed a double quote phrase within a string, e.g., “sample string \"embedded quote\" sample string”
- Backslash can also be used to insert literal new-line and carriage-return symbols using the \n and \r notation, e.g., “sample string\n with new line in it”
- Strings can be concatenated, e.g., “string 1”+“+”string2” produces “string1 string2”
- String themselves or string expressions cannot span multiple lines, i.e., while embedded \n is OK, the actual new line is not.

5.2 Integer Expressions

- Numbers can have the unary minus operator, e.g., -2
- The four arithmetic operations and parentheses, including nested ones, are supported, e.g., 
  \((2+3)*)((7-1)/2+1)\)
- Division by zero produces error in the log; the operation result is undefined.
- Strings cannot be mixed in one expression with numbers, e.g., =2 + “string” is invalid

5.3 Floating Point Expressions

- Either the point or exponent can be used, e.g., 2.00 or 2E00 or 20E-1 or .2E1
- The four arithmetic operations and parentheses, including nested ones, are supported
- Mix of floats and integers produces a floating expression
- Division by zero produces error in the log; the operation result is undefined

5.4 Variables

- Scenario variables are accessed using the common $(varname) format and can be used in either integer or string expressions.

5.5 Built-in Functions

- round(floating_number, precision) - rounds the number to the <precision> number of digits after the point. The result is still a floating point number.
- stripnondigits(string) - removes non-digits characters from string, leaving only digits from 0 to 9, * and # symbols. E.g., stripnondigits("123abc456") will return "123456"
- formatduration(duration_in_seconds) – converts duration in specified in seconds into MM:SS or HHH:MM:SS formats. Produces formatted string as output. E.g., formatduration(121) will return "02:01"
- `titlecase(string)` - converts string to title case (each word is capitalized)
- `tostring(integer)` - converts an integer to a string. E.g., `tostring(-2+1)` should return "-1" as a string
- `urlencode(string)` – URL-encodes a string, replacing special characters using the %dd notation. This is a conservative implementation that replaces all characters that are not explicitly in allowed characters.
- `replace(string, search_pattern, replace_pattern, flags)` - performs search and replace in the input string. Returns the string with replacements performed. Parameters:
  - `string` – the input string to be searched
  - `search_pattern` - regular expression pattern to be matched in the input string. The list of supported patterns can be found in the table below. Note the extra \ in escapes. This is necessary in order to allow literal insertions of " and new line symbols.
  - `replace_pattern` - text to insert instead of matched text according to search pattern. \0 - \9 are allowed in replacements
  - `flags` – one or more of the following additional conditions:
    - `i` – ignore case in the search
    - `g` – replace all matches (otherwise only the first match will be replaced)

E.g., `replace("abcdefg","c","z","ig")` produces “abzdefg”

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>^</td>
<td>Match beginning of a buffer</td>
</tr>
<tr>
<td>$</td>
<td>Match end of a buffer</td>
</tr>
<tr>
<td>()</td>
<td>Grouping and substring capturing</td>
</tr>
<tr>
<td>[...]</td>
<td>Match any character from the set</td>
</tr>
<tr>
<td>[^...]</td>
<td>Match any character but the ones from the set</td>
</tr>
<tr>
<td>\s</td>
<td>Match whitespace</td>
</tr>
<tr>
<td>\S</td>
<td>Match non-whitespace</td>
</tr>
<tr>
<td>\d</td>
<td>Match decimal digit</td>
</tr>
<tr>
<td>\D</td>
<td>Match anything but decimal digit</td>
</tr>
<tr>
<td>\r</td>
<td>Match carriage return</td>
</tr>
<tr>
<td>\n</td>
<td>Match new line</td>
</tr>
<tr>
<td>+</td>
<td>Match one or more times (greedy)</td>
</tr>
<tr>
<td>+?</td>
<td>Match one or more times (non-greedy)</td>
</tr>
<tr>
<td>*</td>
<td>Match zero or more times (greedy)</td>
</tr>
<tr>
<td>*?</td>
<td>Match zero or more times (non-greedy)</td>
</tr>
<tr>
<td>?</td>
<td>Match zero or once</td>
</tr>
<tr>
<td>\meta</td>
<td>Match one of the meta characters: ^$().[+?\</td>
</tr>
</tbody>
</table>
6 Voice Segment Types

This section describes the types of voice segments that can be used to create voice prompts for Collect Digits, Menu, and Play Prompt scenario blocks. The drop-down menu of voice segment types appears when you select Add New in the Prompt list dialog and click Add voice segment. Voice segment parameters depend on the selected segment type and are explained in the table below.

<table>
<thead>
<tr>
<th>Voice Segment Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voice</td>
<td>Plays the content of the uploaded voice file. When creating a new segment of this type, click upload and select the voice file to be played. You can also enter the content of this message as a text in the Content field. If the scenario cannot find the file, it will convert to speech and play the text specified in this field.</td>
</tr>
<tr>
<td>Shared Voice</td>
<td>Plays the content of the selected shared voice segment defined at the contact center level for use in multiple scenarios. For more information, see ServicePattern Contact Center Administrator Guide, section Shared Voice Segments. Select the name of the desired shared voice segment from the drop-down menu.</td>
</tr>
<tr>
<td>Voice, from parameter</td>
<td>Plays the voice segment defined as an external parameter whose value depends on which dial-in scenario entry this scenario is used in. For more information, see ServicePattern Contact Center Administrator Guide, section Dial-in Scenario Entries. Define the variable name here and set its values in property Scenario parameters of the dial-in scenario entries where this scenario will be used. Note that although the full format of the variable is $(app.custom.XXX), you only need to define the XXX part in the variable field.</td>
</tr>
<tr>
<td>Number</td>
<td>Plays the specified number. Can be defined as a variable. Use Frac.digits to define the number of digits after the decimal point to which the natural numbers will be rounded when played as voice. E.g., if Frac.digits is set to 2 and the variable defined in the Value field returns 173.2534895, the voice segment will be played as one hundred seventy-three point twenty-five.</td>
</tr>
<tr>
<td>Ordinal</td>
<td>Plays the specified number as an ordinal number. Can be defined as a variable.</td>
</tr>
<tr>
<td>Currency</td>
<td>Plays the specified amount and the selected currency, e.g. sixty-nine euros. The amount can be defined as a variable. Note that not all currencies are supported in all languages.</td>
</tr>
<tr>
<td>DateTime</td>
<td>Plays the specified date and/or time in the selected format for the selected time zone. Date and time can be defined as a variable.</td>
</tr>
<tr>
<td>Phone</td>
<td>Plays the specified phone number. The number will be pronounced according to the pattern accepted in the selected language. Can be defined as a variable.</td>
</tr>
<tr>
<td>Spell</td>
<td>Spells the specified phrase. Can be defined as a variable. Select the Read Capitals checkbox if the capital letters in the phrase shall be preceded with the word capital. E.g., aZf will be spelled as a-z-f when the checkbox unselected and a-capital-z-f when the checkbox is selected.</td>
</tr>
<tr>
<td>Word</td>
<td>Plays a single-word .wav file with the specified name from the &lt;root&gt;\audio\talkers[selected_language]\words directory. The name can be defined as a variable.</td>
</tr>
</tbody>
</table>

This segment type is currently supported for Enterprise installations only.
<table>
<thead>
<tr>
<th>Voice Segment Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Personal Name</strong></td>
<td>Plays the specified parts of personal names with the correct intonation. E.g., for a full 3-part name the intonation will rise while playing the first name, be neutral for the middle name, and will drop for the last name. The name parts are contained in the <code>&lt;root&gt;/audio/talkers/[selected_language]/names</code> directory. This segment type is currently supported for Enterprise installations only.</td>
</tr>
<tr>
<td><strong>Text</strong></td>
<td>Converts to speech and plays the specified text. Can be defined as a variable.</td>
</tr>
<tr>
<td><strong>URL</strong></td>
<td>Plays the voice content from the source indicated by the specified URL.</td>
</tr>
<tr>
<td><strong>EWT</strong></td>
<td>Plays the current estimated wait time for the given interaction in the <em>hours, minutes, seconds</em> format.</td>
</tr>
</tbody>
</table>
7 Scenario Example

This section provides a simplified example of a typical scenario for processing of inbound service calls. Imagine contact center operations of a security equipment company called All Safe. The contact center provides two services: one for product sales and the other for support. Both services are provided within typical business hours and are available in two languages, English and Spanish. The company’s contact center has one general service number and uses interactive voice response for language and service selection before distributing calls to qualified agents. The following resources are configured in the system to support the above operation:

- services: Sales for sales calls, Support for support calls, and General for unqualified calls and general inquiries
- an auxiliary skill group Languages with skills English and Spanish
- a team of agents specializing in product sales, associated with services Sales and General, where all agents have the corresponding default service skills, all have the English skill, and some also have the Spanish skill
- a team of agents specializing in product support, associated with services Support and General, where all agents have the corresponding default service skills, all have the English skill, and some also have the Spanish skill
- an external access number representing the service number to which all service calls will be made
- a schedule that defines the business hours of the contact center

For more information about configuration of any of the above resource types, see ServicePattern Contact Center Administrator Guide.

A simplified scenario for processing of calls made to the above service number may look like this:

1. First the call arrival time is checked against the contact center operational schedule. If the call arrives outside of the business hours, an announcement is played back prompting the caller to call again during the business hours. The call is then disconnected.
2. If the call arrives within the business hours, a general greeting is played first.
3. The caller is prompted to select the language.
4. Based on the caller’s input, the corresponding language skill is set as one of the agent selection criteria and the language of the subsequent voice prompts is set to match the caller’s preference. If the caller does not provide any input within a timeout, the English language is set for both the language skill and voice prompts.
5. The caller is prompted to select the service.
6. Based on the caller’s input, the corresponding service skill is set as one of the agent selection criteria. If the caller does not provide any input within a timeout, the General Service skill is selected.
7. The call is then queued while the system looks for an available agent who has both selected skills.
8. As soon as such an agent becomes available, the system delivers the call to the agent extension. (If other calls are waiting for the same skill set, they are distributed to agents in the order of their arrival.)
9. If the selected agent does not answer, the call is returned to the service queue to wait for the next available agent with the matching skills.

When translated into the ServicePattern scenario language, the above scenario will look like this.
Once the scenario is defined, a dial-in scenario entry will have to be created in the configuration in order to associate this scenario with the corresponding external access number and schedule. For more information about configuration of dial-in scenario entries, see ServicePattern Contact Center Administrator Guide.