Designer User's Guide

Application phases

11/9/2024

Contents

-
- Administrator

Application flows are comprised of sections of common blocks, known as application phases. Learn about the role each phase plays within the Designer application flow.

Related documentation:

-

Each application flow in Designer contains four phases:

You can learn more about each phase in the sections below.

Initialize

This phase initializes application-level user variables and parameters to use when the application executes. The application initializes during this phase.

By default, the following actions take place:

- Initialize and set up user variables.
- Load application run-time parameters from external sources.
- Process interaction properties (for example, ANI and DNIS) and application run-time parameters. System variables or properties may be initialized internally.
- If configured, additional processing that was set up by the user.

Self Service

The **Self Service** phase is the IVR portion of the interaction. This phase hosts blocks that provide automated interaction with the customer via speech, chat, and/or DTMF. It attempts to provide automated service and contain the interaction within an IVR, so there is no need to route the interaction to an agent.

If routing is necessary, this phase collects necessary data from the user through various questions and menus, and then determines how to route the interaction in the next phase, **Assisted Service**.

> **Tip**
>
> To enable voice call recording for the Self Service phase, set the **EnableSSRecording** variable to **true** in the **System Variables** section.

The following are typical actions that take place during the **Self Service** phase:

- Play Messages. These may be pre-recorded audio files or dynamic text spoken using TTS.
- Check business hours and customize logic based on the outcome (for example, take *this* action if we are closed).
- Collect user input.
- Present choices to customers using menus.
- Navigate customers appropriately, based on their responses (segmentation and branches).
- Call external RESTful APIs and fetch data into user variables.
- Update user variables and write ECMAScript expressions.
- Set up and process global commands and hot words.

The **Self Service** phase updates user variables with collected or calculated data. This data is later used by other blocks in the **Self Service** or **Assisted Service** phase.

Interaction processing might complete during the **Self Service** phase. In this scenario, the application control skips the **Assisted Service** phase and proceeds to the **Finalize** phase. For example, if the business hours check determines that the contact center is closed, the corresponding announcement is played to the caller and the call is terminated.

Assisted Service

This phase hosts blocks that route the interaction to a live agent, if necessary.

During the **Assisted Service** phase, the application attempts to route interactions to agents. Routing is performed based on data collected in previous phases. For example, target skills are taken from user variables.

The following are typical actions for this phase:

- Attempt to route the call while playing music or prompts.
- Call external RESTful APIs.
- Update user variables.

There may be multiple **Route Call** blocks in sequence. Each **Route Call** block might try to route the interaction to different targets with different timeouts. For example, it might expand a target by geographical location.

Each **Route Call** block has a timeout, after which the next **Route Call** block in sequence is executed. If any of the blocks successfully routes a call, the **Assisted Service** phase is complete and processing continues to the **Finalize** phase.

Finalize

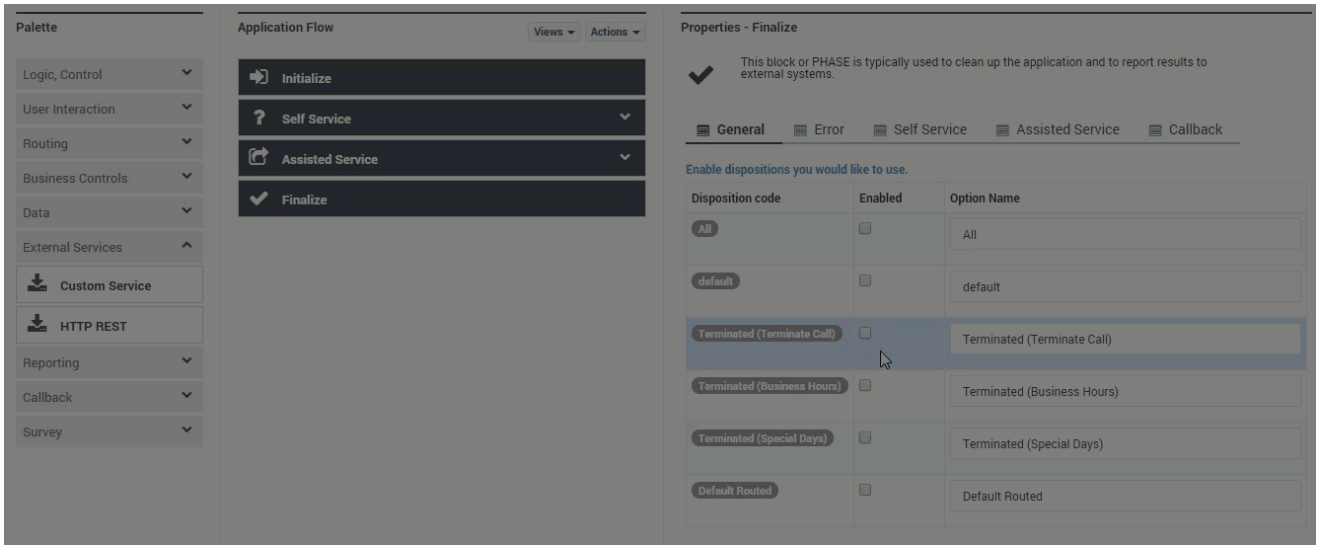
This phase provides post-processing and interaction termination after the customer has been serviced. When interaction processing is finished, the application goes to the **Finalize** phase to perform post-processing for various scenarios that are based on how the interaction was completed.

The following are examples of typical scenarios:

- Interaction was abandoned by the customer (while in either the **Self Service** or **Assisted Service** phase).
- Interaction was completed in **Self Service** phase.
- Interaction was routed to an agent in the **Assisted Service** phase.
- Interaction was delivered to voicemail in the **Assisted Service** phase.
- User opted to leave a queue and schedule a callback.

You can also use the **Finalize** phase to submit application data to an external system for reporting metrics, or to select a Application phases for post-processing. When you click on the **Finalize** block in the application flow, each of the tabs has a list of dispositions that you can select. When you select a disposition, a block for it is created below the main **Finalize** block. You can then drag other valid blocks (such as an HTTP REST block) below the disposition block to further customize the handling for that disposition.

Here's an example:



> ### Tip
> Setting up handlers for the **Finalize** phase is optional. You might not need to do anything special for these cases.

Dispositions

This section describes the dispositions that can be selected in the Finalize phase.

General

Disposition	Description
All	When an application enters the **Finalize** phase, it has only one disposition code, so only one disposition block is executed. However, the **All** disposition code is unique in that it is always executed, in addition to (and after) any disposition block related to the actual disposition code of the application. This is the only case where more than one disposition block is executed. Typically, you would select the **All** disposition code when you want to execute some post-processing logic, no matter what the actual application disposition code is. This is more efficient than duplicating the same logic in every possible disposition block.
Default	This code is used when no other disposition code is applicable. For example, it could indicate that an interaction was not routed, not terminated (by any

Disposition	Description
	party), and was likely still active when the session and/or application completed.
Revoked	The Designer application can no longer control the interaction due to external reasons unrelated to the application logic or processing.
Terminated (Terminate Call)	The application reached a Terminate Call Block.
Terminated (Business Hours)	The interaction arrived outside of regular business hours, as specified in the Business Hours Block.
Terminated (Special Days)	The interaction arrived on a special day, as specified in the Special Day Block.
Terminated (Auto-Stop)	The interaction was processed multiple times by the Designer application and exceeded the defined threshold, or a single application session lasted beyond the **MaxTime** defined in the application (see System Variables). The session and interaction were auto-terminated.
Default Routed	The interaction was delivered to the default routing destination.
Interaction Less	Used when a consult interaction is merged with a parent interaction that the application session is not able to take ownership of. Without an interaction to associate with the session, the application applies this disposition code and proceeds to the **Finalize** phase.

Error

Disposition	Description
System Error	There was an unexpected error in the application (such as a script validation error).

Self Service

Disposition	Description
Abandoned in Self Service	The customer hung up before completing the **Self Service** phase of the application.
Completed in Self Service	The customer successfully completed their interaction in the **Self Service** phase — the application did not need to go to **Assisted Service** and went directly to **Finalize**. If the interaction ended in **Self Service** due to some other condition or event (such as the interaction being received outside of business hours or the customer hanging up), then the appropriate disposition code for that condition or event is used (**Abandoned in Self Service**, **Terminated**, and so on).
Terminated (Emergency)	The emergency flag was set, as specified in the

Disposition	Description
	Emergency Block.
Terminated (Menu Option)	The customer chose a menu option to exit or end the interaction.

Assisted Service

Disposition	Description
Abandoned in Queue	The customer completed the **Self Service** phase, but was disconnected while waiting to speak with an agent.
Routed to Agent	The interaction was successfully delivered to an agent.
Routed to DN	The interaction was successfully delivered to a Direct Number.
Routing Incomplete	The interaction was not delivered to the target destination.
Routed to Voicemail	The interaction was delivered to voicemail.
Routed to Queue	The interaction was delivered to a virtual queue.
Routed to Parking Queue	The interaction was sent to a "parking" queue to wait until the business is open.

Callback

The following final dispositions can be set by the Callback V2 block. However, this disposition value can be overwritten by any subsequent block that the Designer application enters.

Disposition	Description
Callback Not Offered	Callback was not offered to the customer.
Callback Accepted	The customer accepted the callback offer.
Callback Declined	The customer declined the callback offer.
Callback Cancelled	The callback was cancelled.
Callback Rescheduled	The callback was rescheduled.
Callback Abandoned in Queue	The customer was reconnected, but abandoned the interaction queue while waiting for an agent.
Callback Routed to Agent	The customer was reconnected and routed to an agent.
Callback Outbound Failed	The customer was not successfully reconnected after the maximum number of attempts.
Callback Keep Existing	The customer had already booked a callback. When they called back in, they were informed they already had a callback. They chose to keep their existing callback and disconnected.
Callback Push Failed	The callback Push Notification failed to be sent to

Disposition	Description
	the customer's device.
Callback Booking Failed	The callback failed to be booked.
Callback Purged	The callback was purged from the system.
Callback Matched	The customer called-in and was matched to an existing callback request.
Callback Push Expired	The Push Notification expired before the customer responded to it.
Callback Push Delivered	The Push Notification was successfully delivered to the customer's device.
Callback Call Moved	The callback was moved to another session.