Agent Interaction SDK Services Developer Guide

The Interaction Service

12/14/2025

Contents

The Interaction Service

The interaction service is defined by the `IInteractionService` interface defined in the`com.genesyslab.ail.ws.interaction` namespace. Its features are designed to manage characteristics common to all interactions, no matter their media.

Introduction

This section introduces the interaction concepts and the interaction service covered in this chapter.

What Is an Interaction?

An interaction is the representation of the communication between two persons or more in the Genesys world. It can represent, for example, a phone call between an agent and a customer, an agent and an incoming e-mail, or a chat session between agents and a customer.
An interaction exists:

- From the moment that a contact gets in touch with the agent (a call or an e-mail is ringing on the agent desktop) or from the moment that the agent initiates getting in touch with a contact or another agent.
- Till the interaction is done: the agent hangs up (or closes the e-mail) and marks the interaction as done.

Interactions are associated with a particular medium:

- Voice interactions (or phone calls) are associated with DNs.
- E-mail (or mail) interactions are associated with the `EMAIL` media.
- Chat interactions are associated with the `CHAT` media.

What Is the Interaction Service?

Although the underlying media are different, interactions share some common characteristics and similar data management.
The interaction service is an interface designed to read and write DTOs containing:

- Common attributes defined in the `IInteractionService` interface.
- Interaction-specific attributes defined in other interactions interfaces such as `IInteractionVoiceService` and `IInteractionMailService` .

The `IInteractionService` interface provides dedicated DTO methods using DTO classes of the `com.genesyslab.ail.ws.interaction` namespace. See also Data Transfer Object for details.
The following are examples of common interaction attributes defined in the interaction service:

- `interaction:interactionId`—the system interaction identifier.
- `interaction:contactId`—the contact identifier.

- `interaction:status`—the interaction status defined with the `InteractionStatus` enumeration.
- `interaction:interactionType`—the interaction type defined with the `InteractionType` enumeration.
- `interaction:eventReason`—the `InteractionEventReason` value published with interaction events.
- `interaction:outboundChainId`—if an interaction arrives, and it is in an outbound context, this attribute is filled with the corresponding outbound chain ID. See The Outbound Service for details.

The interaction service does not let you perform any actions on an interaction, such as making a call or sending an e-mail. You may have noticed, that there is no existing interaction action enumeration in the `com.genesyslab.ail.ws.interaction` namespace. However, the interaction service has methods for adding, modifying, and deleting attached data of any existing interaction (see Attached Data).
The interaction service also offers `InteractionEvent`, that are generic interaction events used by specialized interaction services such as `IInteractionVoiceService` and `IInteractionMailService`.

Specific Interaction Services

The Agent Interaction Service API includes other interaction services and their associated namespaces:

- `IInteractionVoiceService` (`com.genesyslab.ail.ws.interaction.voice`)—this service is dedicated to voice interactions occurring on the DNs of a Place. This service requests actions on voice interactions.
- `IInteractionMailService` (`com.genesyslab.ail.ws.interaction.mail`)—this service is dedicated to e-mail interactions occurring on the `EMAIL` media of a Place. This service requests actions on e-mail interactions.
- `IInteractionChatService` (`com.genesyslab.ail.ws.interaction.chat`)—this service is dedicated to chat interactions occurring on the `CHAT` media of a Place. This service requests actions on chat interactions.

These services deal with specialized interaction actions, but to properly use them, your application must rely on the interaction service for interaction data handling and integrate `InteractionEvent` events. The following sections present details.

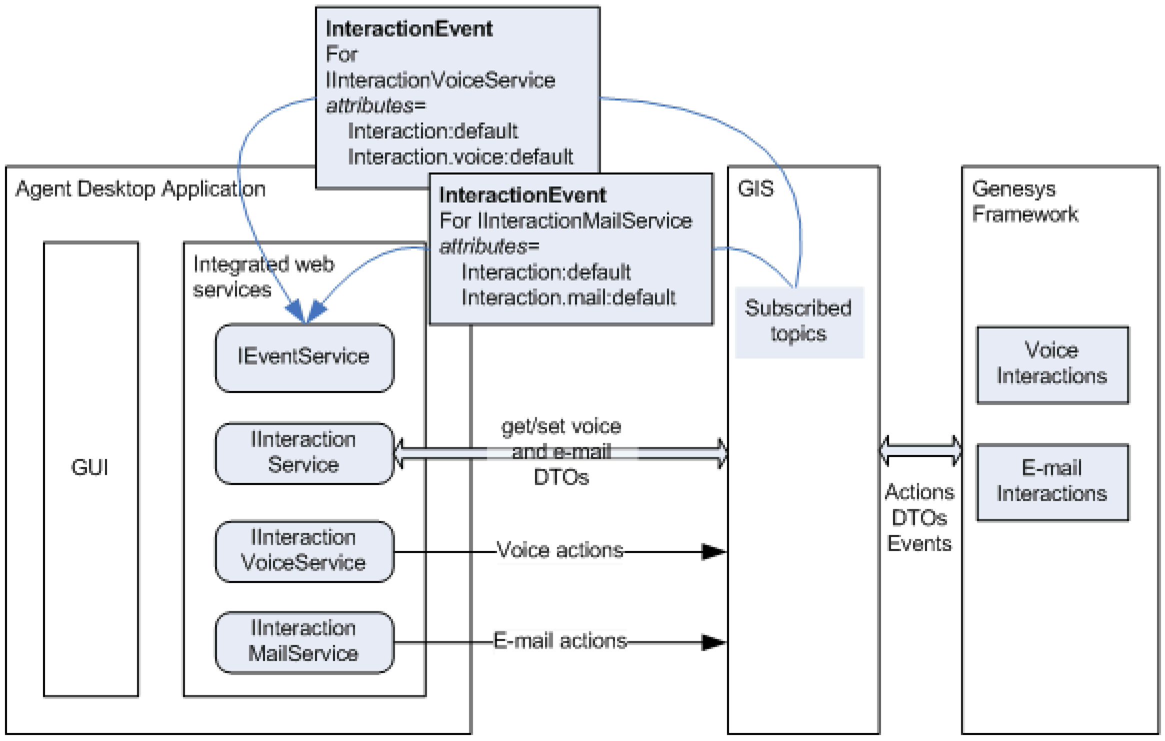
Using IInteractionService

The `IInteractionService` interface is used to manage interactions' data that correspond to attributes defined in the other interactions services interfaces. Your application is likely to use it any time it needs to access and modify interactions' data.
Moreover, the `IInteractionService`interface offers `InteractionEvent` which are generic events publishing the following attributes having theevent property:

- `interaction:*`—`IInteractionService` attributes
- `interaction.*:*`—attributes for e-mail, chat, and voice interaction services.

`InteractionEvent` events are the most common events received by your application after the media-specialized interaction services have successfully performed actions.
To properly deal with interactions and specialized services, your application has to subscribe to these

events. The following figure illustrates relationships between services, interactions, and `InteractionEvent` received by the `IEventService`.



Interactions and Services

This figure shows that the `IInteractionService` interface manages all data transactions and that specialized services—e-mail and voice services—are used to perform dedicated actions on the corresponding interactions.
The `IEventService` has subscribed to `InteractionEvent` with `TopicsEvent` having a trigger on the agent identifier (or place identifier). The propagated attributes include some `IInteractionService`, `IInteractionVoiceService`, and `IInteractionMailService` attributes.
When an `InteractionEvent` occurs:

- If the `Event` object retrieved is related to a voice interaction, it propagates `IInteractionService` and `IInteractionVoiceService` attributes.
- If the `Event` object retrieved is related to an e-mail interaction, it propagates `IInteractionService` and `IInteractionMailService` attributes.

Important

The media-specialized services interfaces may offer other specific events. For further information, see the Agent Interaction SDK 7.6 Services API Reference.

Handling Interaction DTOs

The `IInteractionService` interface has two general methods to deal with DTOs:

- `IInteractionService.getInteractionsDTO()`—This method retrieves the DTOs of a set of interactions for a set of attributes having the read property.
- `IInteractionService.setInteractionsDTO()`—This method uses DTOs to set the values of a set of interactions for a set of attributes having the `write` property.

> **Important**
> These methods offer standard DTO handling. See Data Transfer Object

After your application gets an instance of the `IInteractionService` interface, it can use the above methods or specific DTO methods detailed in the following subsections.

Specific Getting DTO Methods

The `IInteractionService` interface retrieves attributes of any interaction type with interactions DTOs according to their associated agents, place identifiers, or DN identifiers.

- The `IInteractionService.getInteractionsDTOFromAgent()` method retrieves DTOs for voice, chat, or e-mail interactions associated with agents.
- The `IInteractionService.getInteractionsDTOFromPlace()` method retrieves DTOs for voice, chat, or e-mail interactions associated with a place if an agent is logged on this place.
- The `IInteractionService.getInteractionsDTOFromDN()` method retrieves DTOs for voice, chat, or e-mail interactions associated with a DN if an agent is logged on this DN.

> **Important**
> You can also use the IInteractionService.getInteractionsDTO() method to retrieve the DTOs of a set of interactions.

The following code snippet shows how to use `IInteractionService.getInteractionsDTOFromAgent()` method to retrieve the interaction identifiers and status available for `Agent0`.

```
InteractionAgentDTO[] ixnAgentDTOs = myInteractionService.getInteractionsDTOFromAgent( new
string[]{"agent0"},
new string[]{ "interaction:interactionId", "interaction:status"});

foreach(InteractionAgentDTO agentDTO in ixnAgentDTOs)
{
   System.Console.WriteLine(“Agent: “+ agentDTO.agentId+"\n");
   foreach(InteractionDTO dto in agentDTO.interactionsDTO)
   {
      System.Console.WriteLine(dto.interactionId+" ");
      foreach(KeyValue attributes in dto.data)
      {
         System.Console.WriteLine(attributes.key+": " +attributes.value.ToString()+"\n");
      }
   }
}
```

Opening a Workbin Interaction

The `IInteractionService` interface has two general methods to opens a workbin interaction:

- `IInteractionService.openInteractionForAgentDTO()`—This method opens a workbin interaction for an agent and a set of attributes.
- `IInteractionService.openInteractionForPlaceDTO()`—This method opens a workbin interaction for a place and a set of attributes. Once the interaction is opened, it goes onto the specified place for treatment.

The following code snippet shows how to use `IInteractionService.openInteractionForAgentDTO()` method to open a workbin interaction for `Agent0`.

```
InteractionDTO InteractionAttributes =
   myInteractionService.openInteractionForAgentDTO(
      "agent0", // the agent Id
      myInteractionId, // the interaction ID
      new string[]{
      "interaction:interactionId",
      "interaction:status"
   } // the set of the interaction’s attributes
);
```

Attached Data

Attached Data—also known as User Data—is a set of key-value pairs attached to an interaction and stored with the interaction in the contact history. Attached Data can be collected by the Genesys Solution and modified by the agent.
An attached data can be any data useful to your application’s design. However, it can also include the following specific attached data:

- Interaction attribute values.
- Custom attached data’s values.

The Configuration Layer defines keys and information for these attached data, available through the `IResourceService` interface. See Interaction Information for further details.

Attached Data DTOs

Attached Data are handled with the following attributes of the `IInteractionService` interface:

- `interaction:attachedData`—This attribute is used to read or write the attached data of the interaction.

> **Important**
> Writing with these DTOs replaces the previous attached data by the ones written.

- `interaction:addAttachedData`—This attribute is used to write the added attached data in the interaction.
- `interaction:removeAttachedData`—This attribute is used to remove all the attached data of a set of interactions.

The `com.genesyslab.ail.ws.interaction` namespace includes the `AttachedData` container class to deal with DTOs. An `AttachedData` object represents a single key-value pair attached to an interaction.
Use the `IInteractionService.setInteractionsDTO()` to write DTOs and the `IInteractionService.getInteractionsDTO*()` to read them.
In the following code snippet, a new attached data is added to the `Interaction0` interaction:

```
/// Creating a AttachedData
AttachedData myData = new AttachedData();
myData.key = "Key0";
myData.value = "This is the value of Key0";

/// Creating an array of InteractionDTO
InteractionDTO[] ixnDTOs = new InteractionDTO[1];

/// Filling the first DTO
ixnDTOs[0] = new InteractionDTO();

/// Specifying the concerned Interaction
ixnDTOs[0].interactionId = “Interaction0”;

/// Creating an array of DTOs
ixnDTOs[0].data = new KeyValue[1];
ixnDTOs[0].data[0] = new KeyValue();

/// Giving the DTO content
ixnDTOs[0].data[0].key = "interaction:addAttachedData";
ixnDTOs[0].data[0].value = new AttachedData[]{ myData };

/// Writing the DTOs with the IInteractionService
myInteractionService.setInteractionsDTO(ixnDTOs);
```

Attached Data and Event

When the attached data of an interaction are modified, an `InteractionEvent` is sent with an`InteractionEventReason.INFO_CHANGED` reason for the `interaction:eventReason` attribute. The modified attached data are available in the `interaction:modifiedAttachedData` attribute.