Orchestration Server Deployment Guide

SCXML and ORS Extensions

12/12/2025

Contents

SCXML and ORS Extensions

Orchestration applications are created by writing SCXML documents either via your favorite text editor or via Genesys Composer. Orchestration-specific instructions are specified in the executable content of SCXML in the form of SCXML extensions (action elements) and/or ECMAScript extensions (properties of special ECMAScript objects). See SCXML Language Reference in the Orchestration Server Developer's Guide.

ORS Extensions to SCXML

ORS 8.1 extensions to the SCXML executable content are described in Orchestration Extensions available in the Orchestration Server Developer's Guide. The example below uses the Queue module. The namespace definition of the Queue module is the following:

```
xmlns:queue="www.genesyslab.com/modules/queue"
```

The Queue module can be called/addressed within the logic of the application as follows:

```
<queue:submit queue="vq1" priority="5" timeout="100">
```

ORS 8.1 provides the modules listed below.

- **Classification module**. This module element uses the namespace label classification which stands for www.genesyslab.com/modules/classification. It implements the ability to classify non-voice interactions such as e-mail, chat and open media by a given category and screen content of non-voice interaction by a given set of screen rules.
- **Queue module**. This module element uses the namespace label queue which stands for www.genesyslab.com/modules/queue. It implements the target selection functionality of URS (finding resources for interactions and delivering interactions to the resource).
- **Dialog module**. This module element uses the namespace label dialog which stands for www.genesyslab.com/modules/dialog. It implements the call treatment functionality.
- **Web Services module**. This module element uses the namespace label ws, which stands for www.genesyslab.com/modules/ws. This module provides Web Services support in Orchestration, and covers both Web 2.0 RESTful Web Services interface and legacy SOAP Web Services interface.
- **Statistics module**. This module element uses the namespace label statistic, which stands for www.genesyslab.com/modules/statistic. It implements the statistics retrieving functionality of URS.
- **Interaction module**. This module element uses the namespace label ixn, which stands for www.genesyslab.com/modules/interaction. It implements getting and changing interaction related data, such as attached data, and makes calls, transfers, conferences, and performs other related functions.
- **Session module**. This module element uses the namespace label session which stands for www.genesyslab.com/modules/session. The module maintains common objects used by the Orchestration Platform for Orchestration logic reporting and management functionality.

- **Resource module**. This module element uses the namespace label resource which stands for www.genesyslab.com/modules/resource. This module maintains a common entity (called a resource) that is used across functional module interfaces.

Action Elements

Developers specify action items as custom action elements inside SCXML executable content. All modules are prefixed with the corresponding namespace label; for example:

`queue:submit` (in this case, the queue prefix stands for www.genesyslab.com/modules/queue)

`dialog:playandcollect` (in this case, the dialog prefix stands for www.genesyslab.com/modules/dialog)

Executing Modules

When modules are executed, events return back from the platform to the instance of logic that is running the SCXML document that requested the action.

Functions and data are exposed (and used) as properties of ECMAScript objects. ORS provides a built-in ECMA Script object for every module listed above.