



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

Orchestration Server Developer's Guide

Change the Ownership of an Interaction

12/16/2025

Change the Ownership of an Interaction

The following example illustrates how to detach an interaction (ex. a voice call) from the parent session, and attach it to a child session that was started using `<session:start>`. The child session operates completely independently of the parent session, meaning that the termination of the parent session has no effect on the child session, and vice versa. Also, session content is not shared between the sessions.

Parent strategy

The following SCXML application detaches an interaction, then starts a new child session. The session terminates when we get a confirmation that the child session has been started.

```
<scxml version="1.0" xmlns="http://www.w3.org/2005/07/scxml"
  xmlns:queue="http://www.genesyslab.com/modules/queue"
  xmlns:ixn="http://www.genesyslab.com/modules/interaction"
  xmlns:session="http://www.genesyslab.com/modules/session"
  initial='initial'>
  <datamodel>
    <data id="reqid" expr="" />
    <data id="ixnid" expr="" />
    <data id="newsessionid" expr="" />
  </datamodel>
  <state id="initial">
    <transition event="interaction.added" target="detachcall">
      <script>
        _data.ixnid = _event.data.interactionid;
      </script>
    </transition>
  </state>
  <state id="detachcall">
    <onentry>
      <log expr="'Detach call from current session ...'" />
      <ixn:detach requestid="_data.reqid" interactionid="_data.ixnid" />
    </onentry>
    <transition event="error.interaction.detach" target="error">
      <log expr="'Got detach error:'" />
      <log expr="uneval( _event )" />
    </transition>
    <transition event="interaction.detach.done"
      cond="_event.data.requestid == _data.reqid &&
      _event.data.interactionid==_data.ixnid"
      target="create_new_session">
      <log expr="'Got detach confirmation'" />
      <log expr="uneval( _event )" />
    </transition>
  </state>
  <state id="create_new_session">
    <onentry>
      <log expr="'Starting a new session that will attach the interaction
      ...'" />
      <session:start src="'http://localhost:17080/ixn_attach.scxml'"
        sessionid="_data.newsessionid">
        <param name="parent_sessionid" expr="_sessionid" />

```

```
        <param name="interactionid" expr="_data.ixnid" />
    </session:start>
</onentry>
<transition event="session.start.done" target="exit">
    <log expr="'**** Start Session Done'" />
    <log expr="'**** Event details: ' + uneval( _event )" />
    <log expr="'**** New Session ID: ' + _data.newsessionid" />
</transition>
<transition event="error.session.start" target="error">
    <log expr="'Start Session Error'" />
    <log expr="uneval( _event )" />
</transition>
</state>
<final id="exit" />
<final id="error" />
</scxml>
```

Child strategy

The following SCXML application was used by the parent session to start the child session. Two parameters were passed in from the parent session, `parent_sessionid` and `interactionid` (the `interactionid` of the voice call that was detached from the parent session). Using the `interactionid`, the child session is able to do an attach and take ownership of the interaction. Then we accept the call on DN 702.

```
<scxml version="1.0" xmlns="http://www.w3.org/2005/07/scxml"
  xmlns:queue="http://www.genesyslab.com/modules/queue"
  xmlns:ixn="http://www.genesyslab.com/modules/interaction"
  xmlns:session="http://www.genesyslab.com/modules/session"
  initial='initial'>
  <datamodel>
    <data id="reqid" expr="'" />
    <data id="ixnid" expr="'" />
    <data id="newsessionid" expr="'" />
    <data id="DN2" expr="'702'" />
    <!-- Parameters passed in from parent session
    parent_sessionid
    interactionid
    -->
  </datamodel>
  <state id="initial">
    <onentry>
      <log expr="'New session has been started ...'" />
      <log expr="'Datamodel = ' + uneval(_data)" />
    </onentry>
    <transition target="attachcall" />
  </state>
  <state id="attachcall">
    <onentry>
      <log expr="'Attach call to this new session ...'" />
      <ixn:attach requestid="_data.reqid"
interactionid="_data.interactionid" />
    </onentry>
    <transition event="error.interaction.attach" target="error">
      <log expr="'Got attach error:'" />
      <log expr="uneval( _event )" />
    </transition>
    <transition event="interaction.attach.done"
      cond="_event.data.requestid == _data.reqid && _event.data.interactionid
```

```
==_data.interactionid"
    target="acceptcall">
        <log expr="'Got attach confirmation:'" />
        <log expr="uneval( _event )" />
    </transition>
</state>
<state id="acceptcall">
    <onentry>
        <log expr="'Answering call on dn = ' + _data.DN2" />
        <ixn:accept requestid="_data.reqid"
interactionid="_data.interactionid"
        resource="_data.DN2" />
    </onentry>
    <transition event="error.interaction.accept" cond="_event.data.requestid ==
_data.reqid"
        target="error">
            <log expr="'*** Got accept error:'" />
            <log expr="uneval( _event )" />
        </transition>
        <transition event="interaction.accept.done"
            cond="_event.data.requestid == _data.reqid && _event.data.interactionid ==
_data.interactionid"
            target="exit">
                <log expr="'*** Got accept confirmation on DN = ' + _data.DN2" />
                <log expr="'event details = ' + uneval( _event )" />
            </transition>
    </state>
    <final id="exit" />
    <final id="error" />
</scxml>
```