Call Start Action

VXML Server can be configured to run code when a call has been received before the call flow is visited. The call start action can be implemented with either the Java API or the XML API. The call start action is a good way to create session data to be used by the rest of the application. There are two situations where session data may already exist:

- If the voice browser passed additional arguments to VXML Server when the call was first received, these additional arguments are added as session data with the arguments' name/value pairs translated to the session data name and value (each is a `String`).
- If a separate Unified CVP voice application transferred to the current application, the application designer may have chosen to transfer element and session data to the destination application. This data will be converted to session data in the destination application.

The call start action is also given the ability to change the voice browser and any root document-affecting settings for the call. These changes apply to the current call only, and allows for a truly dynamic application. By allowing the voice browser to change, the application can be deployed on multiple voice browsers at once and use a simple DNIS check to output VoiceXML to the appropriate browser. Changing root document settings such as properties and language allow the call start action to control how the application appears to the caller using information it knows only at call time.

Note These changes can only be made by the call start action because it runs before VXML Server has returned the first VoiceXML page and therefore can make changes that affect the outputted VoiceXML. Aside from these settings, the call start action can also change the maintainer and default audio path, though any component run within the call can do this as well.

The start of call event can be run in the background by checking the appropriate checkbox in the Call Studio application settings. If this is not done, the caller will hear silence until the call start action is complete and the call flow reaches the first VoiceXML-producing element. Answering the phone with too much silence could cause the caller to end the call, thinking something went wrong. Latency issues are not as big a concern later in the application because audio can be played while action is run or the application could make the caller aware that some potentially lengthy action is about to occur. Running the call start action in the background will ensure that the call flow will begin immediately.

Some notes of caution are warranted when running the call start action in the background:

- Ensure that elements in the call flow that attempt to access data created by the call start action do not try too quickly since it is possible the data has not been created yet. Since the call start action is run in a

separate thread, there is no guarantee it will complete before the data it creates is required. The application can be architected to handle this by checking if the data exists before accessing it and if not, make the caller wait until it is created.

- Any errors that occur while running this action are placed in the error log but do *not* end the call (unless the application cannot run without performing the tasks in the call start action).

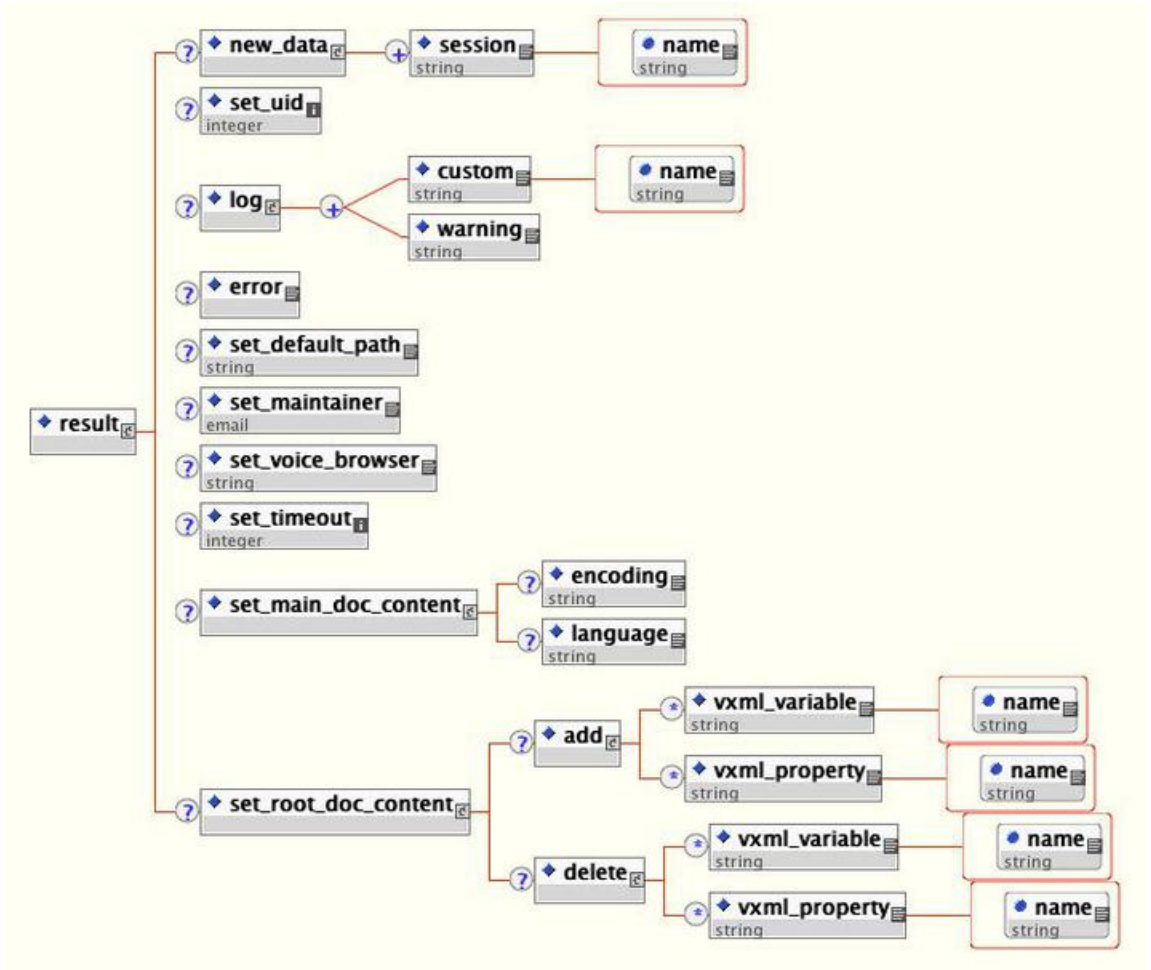
Java API Use

The call start action is built in Java by implementing the Unified CVP class `StartCallInterface` found in the `com.audium.server.proxy` package. It contains a single method named `onStartCall` that is the method for the call start class. This method receives a single argument, an instance of `CallStartAPI`. This class belongs to the Session API and is used to access and modify session information such as session data (See Session API for more information on this API). The method does not have a return value. It is expected that, should an unrecoverable error occur, the call start action will throw an `AudiumException`.

XML API Use

As described in Session API, the standard *inputs* and *settings* XML documents are sent via POST to the call start URI. The following figure shows the DTD diagram of the XML document that must be sent in response. The DTD for the start of call action response is defined in the file `CallStartResponse.dtd` found in the VXML Server `dtds` folder.

Figure 1: DTD Diagram of XML Document to Be Sent in Response



The tags in these XML documents are:

- **new_data** – This tag holds the session data to be created. Any number of `<session>` tags can appear, one for each session data variable to be created.

Note Element data cannot be created because the call start action is not an element.

- **set_uid** – This tag is used to associate the call to a UID in the user management system. The content of the tag should be the integer UID.
- **log** – This tag is used to trigger logger events for this application. Any number of `<custom>` tags can appear, denoting the triggering of a custom event. The `name` attribute holds the name of the data, and the `<custom>` tag encapsulates the value. Any number of `<warning>` tags can appear, denoting the triggering of a warning event. The `<warning>` tag encapsulates the warning message.
- **error** – This tag reports to VXML Server that an error occurred while running the call start action. VXML Server will then throw an exception whose message is contained in the `<error>` tag. This allows the XML API to throw exceptions just as the Java API does.

- **set_default_path** – This tag is used to change the default audio path.
- **set_maintainer** – This tag is used to change the maintainer e-mail address.
- **set_voice_browser** – This tag is used to change the voice browser for this particular call.

Note The real name of the voice browser must be used here, *not* the display name. Gateway Adapter real names can be seen by reading the folder name for that adapter in the `gateways` folder of VXML Server.

- **set_timeout** – This tag allows the timeout length set for this session to be changed. The contents of the tab must be an integer representing the number of minutes in the timeout.
- **set_main_doc_content** – This tag allows the encoding and language settings for the application to be changed for this call. The `<language>` tag content is formatted according to the specification for using languages in VoiceXML (for example, *en-US*). The `<encoding>` tag content is formatted according to the specification for encoding XML pages (for example, *UTF-8*).
- **set_root_doc_content** – This tag allows the addition and removal of VoiceXML properties and variables. The `<delete>` tag is necessary only if properties or variables were set in the application's project pane in Call Studio and due to runtime circumstances the call start action determines they are no longer needed. The `name` attribute specifies the property or variable name and the tag contents encapsulate the value.

Note All the tags are optional, there is no tag required except for the root `<result>` tag. Since the XML API requires a document in response, it is acceptable to return an XML document whose `<result>` tag is empty.

Remote Execution

For remote execution of the Call Start Action, the following syntax for URI is to be used:

For HTTP and RPC call:
`remote://system/?classurl=<fully_qualified_java_class_path>`

For example:
`remote://system/?classurl=com.cisco.cvp.callstudio.Action.TestCallClass`

`remote://system` indicates that the configurations will be fetched from the **Remote Url Settings** property tab which is application-specific.

Note If a direct remote server URI is provided, then that **IP:Port** will be used and not fetched from the **Remote Url Settings** property tab.

For example:
`http://<IP>:<Port>/<target_path>/?classurl=<fully_qualified_java_class_path>`