Platform Certification Program Certification Test Report

VoiceXML versions:	VoiceXML Version 2.0 W3C Recommendation - March 16, 2004 VoiceXML Version 2.1 W3C Recommendation - June 19, 2007
SRGS version:	SRGS Version 1.0 W3C Recommendation - March 16, 2004
Test Suite version:	VoiceXML Forum VoiceXML Certification Test Suite, Version 2.1
Date Certified:	November 29, 2016

Vendor Information

	Vendor Information
Vendor	Interactive Media S.p.A.
Address	Viale Citta' d'Europa 679, 00144 Rome, Italy
Contact:	Claudio Spallaccini
Email:	cspallaccini@imnet.com
Phone:	+39 06 96 700 762
Fax:	+39 06 96 700 763

Implementation Under Test

The following implementation is certified by the VoiceXML Forum as having passed all required assertions and all vendor-supported optional assertions in the modules listed below, except for those optional features designated as omitted.

	Platform Configuration
Implementation Name:	IM VXML NG
Implementation Identifier:	7.2.6

User Agent:	IM VXML_NG/7.2.6
Operating System Version:	Red Hat Enterprise Linux 7.2
Telephony Configuration:	SIP
ASR Engine:	
TTS Engine:	
Platform Placement:	The tested platform was **remote**. The hardware and software tested were **not** in the test lab when tested, and were accessed remotely.

VoiceXML 2.0 Module

Notes

1. Check log files in `/opt/imnet/log`
2. Assertion 629: Ensure ASR is stopped when the test is executed (by directly configuring the platform to execute assertion 629 with ASR stopped or stopping service immediately after the "assertion 629" prompt is played).

Optional assertions

Tested?	ID	Spec. Reference	Abstract	Reason
No	207	[2.2]	When the accept attribute of choice is set to "approximate", or is not set and the accept attribute of its enclosing menu is set to "approximate", then the text of the choice element defines an approximate recognition phrase, as defined under "Grammar Generation" in section 2.2.	
No	239	[2.3.1]	The type attribute, if present, specifies the name of one of the following builtin grammar types from Appendix P: boolean, date, digits, currency, number, phone, time.	
No	250	[2.3.1.2]	A builtin URI of the form "builtin:grammar/<type>" references a builtin speech grammar.	
No	251	[2.3.1.2]	[optional] A builtin URI of the form "builtin:dtmf/<type>" references a builtin DTMF grammar.	
No	252	[2.3.1.2]	[optional] A field containing two builtin grammar elements, one referring to the builtin speech grammar and the other referring to the builtin DTMF grammar of the same type is equivlent to a field whose type attribute is set to the equivalent type.	
Yes	286	[2.3.6]	If the caller is silent for finalsilence milliseconds during a recording, the recording is terminated.	
No	288	[2.3.7]	A VoiceXML document can initiate a transfer to another entity using the transfer tag, such that the Interpreter remains connected to the original caller and interpretation resumes upon termination of the transfer.	
No	289	[2.3.7]	A VoiceXML document can initiate a transfer to another entity using the transfer tag, such that the Interpreter disconnects from the caller immediately upon attempting the transfer and continues execution as it would under termination of the Session.	
No	290	[2.3.7]	A bridged transfer can contain speech grammars such that the interpreter listens to the original caller for the duration of the transfer, terminating the transfer as soon as a spoken utterance matches an active speech grammar.	

No	291	[2.3.7]	During a bridged transfer, any speech grammars outside the transfer element are not active and cannot be matched.	
No	292	[2.3.7]	A transfer element that specifies both a 'dest' and 'destexpr' attribute results in an 'error.badfetch' upon loading of the document containing it.	
Yes	293	[2.3.7]	A transfer element specifying a 'cond' attribute that evaluates to false upon selection of the element by the FIA is not executed.	
No	294	[2.3.7]	A transfer element not specifying a 'bridge' attribute is executed as a blind transfer.	
No	295	[2.3.7]	A bridged transfer specifying a 'connecttimeout' attribute with a W3C time specification will terminate a transfer attempt if the destination entity cannot be connected to within that period of time.	
No	296	[2.3.7]	A bridged transfer specifying a 'maxtime' attribute with a W3C time specification will terminate a transfer after that period of time has elapsed after connecting to the destination entity if it has not already been terminated for other reasons.	
No	297	[2.3.7]	A bridged transfer specifying a 'transferaudio' attribute with valid URI to an audio file wil play that audio from the beginning of the transfer attempt until the attempt is terminated or the destination entity is connected to.	
No	298	[2.3.7]	Upon a successful blind transfer, a 'connection.disconnect.transfer' event is thrown and the transfer name variable remains undefined.	
No	300	[2.3.7]	If the originating caller hangs up during a bridged transfer, a 'connection.disconnect.hangup' event is thrown and the transfer name variable remains undefined.	
No	301	[2.3.7]	If the Interpreter is unable to connect to the destination entity when attempting a transfer because it is busy, the transfer name variable is filled with the value 'busy'.	
No	304	[2.3.7]	If a transfer is terminated because the original caller matches an active DTMF grammar, the transfer name variable is filled with the value 'near_end_disconnect'.	
No	305	[2.3.7.2.2]	If a transfer is terminated because the destination entity disconnects, the transfer name variable is filled with the value 'far_end_disconnect'.	
No	307	[2.3.7.2.2]	If a bridge transfer is terminated by the Interpreter because the 'maxtime' has expired without the transfer being terminated, the transfer name variable is filled with the value 'maxtime_disconnect'.	
No	308	[2.3.7]	Upon termination of bridge transfer, the shadow variable 'name$.duration' (where name is the name attribute of the transfer element) is set to the number of seconds from the time the destination entity was connected to and the transfer was terminated.	
No	309	[2.3.7.2]	Upon termination of bridge transfer due to the caller matching an active speech grammar ('near_end_disconnect'), the 'name$.inputmode' shadow variable is set to 'voice'.	
No	310	[2.3.7]	Upon termination of bridge transfer due to the caller matching an active DTMF grammar ('near_end_disconnect'), the 'name$.inputmode' shadow variable is set to 'dtmf'.	
No	311	[2.3.7]	Upon termination of a bridge transfer due to the caller matching an active speech grammar ('near_end_disconnect'), the 'name$.utterance' shadow variable is set to utterance text.	
No	379	[4.1.5]	When set to false, the bargein attribute on the prompt the user should be unable to interrupt the playback of the contents of the prompt.	
No	380	[4.1.5]	If bargein occurs during any prompt in a sequence, all subsequent prompts are not played. This is required when barge-in is supported by a platform.	

No	382	[4.1.5]	If the bargein attribute is not specified, then the value of the bargein property is used if set.	
Yes	387	[4.1.5]	A nomatch event will never be generated in the case of hotword barge-in.	
No	409	[5.1.4]	session.connection.local.uri returns a URI which addresses the local interpreter context device.	
No	410	[5.1.4]	session.connection.remote.uri returns a URI which addresses the remote caller device.	
No	411	[5.1.4]	session.connection.protocol.name returns the name of the connection protocol.	
No	412	[5.1.4]	session.connection.protocol.version returns the version of the connection protocol.	
No	413	[5.1.4]	session.connection.redirect returns an array representing the connection redirection paths. The first element is the original called number, the last element is the last redirected number. Each element of the array contains a uri, pi (presentation information), si (screening information), and reason property. The reason property can be either "unknown", "user busy", "no reply", "deflection during alerting", "deflection immediate response", "mobile subscriber not reachable".	
No	414	[5.1.4]	session.connection.aai returns an application-to-application information passed during connection setup.	
No	415	[5.1.4]	session.connection.originator returns the local or remote property. (For instance, the following ECMAScript would return true if the remote party initiated the connection: var caller_initiate = connection.originator == connection.remote;	
No	491	[5.2.6]	With universal grammars active, requesting to cancel playing of the current prompt throws the cancel event.	
No	492	[5.2.6]	With universal grammars active, asking to exit throws the exit event.	
No	493	[5.2.6]	With universal grammars active, asking for help throws the help event.	
No	495	[5.2.6]	When the user has been transferred unconditionally to another line and will not return the connection.disconnect.transfer event is thrown.	
Yes	579	[6.1.3]	When an interpreter context does prefetch a resource, it must ensure that the resource fetched is precisely the one needed.	
No	601	[6.3.4]	The value of the bargein property controls whether barge-in is allowed by default for prompts.	
No	624	[2.3.7]	A bridged transfer can contain DTMF grammars such that the interpreter listens to the original caller for the duration of the transfer, terminating the transfer as soon as DTMF input matches an active DTMF grammar.	
No	1015	[2.3.6]	If no audio input is received from the user before the timeout period expires, then record variable is not defined, and a timeout event is thrown.	
No	1021	[2.3.6]	Speech input not matching any active speech grammar is ignored (and therefore does not terminate or otherwise affect recording).	
Yes	1027	[2.3]	The form item object has a result variable, specified by the name attribute. This variable may be given an initial value with the expr attribute.	
Yes	1029	[2.3]	The form item transfer has a result variable, specified by the name attribute. This variable may be given an initial value with the expr attribute.	
Yes	1034	[2.3]	The form item 'object' has a guard condition specified with the cond attribute. A form item is visited if it is not filled and its cond is not specified or evaluates, after conversion to boolean, to true.	
No	1041	[2.3]	The input item transfer may contain the filled element. Filled elements	

			contain an action to execute after the result input item variable is filled in.	
Yes	1046	[2.3]	The input item 'transfer' may contain the property element. Property elements specify properties that are in effect for this input item.	
Yes	1049	[2.3]	The input item 'object' may contain the prompt element. Prompt elements specify prompts to play when visiting this input item.	
No	1051	[2.3]	The input item 'transfer' may contain the prompt element. Prompt elements specify prompts to play when visiting this input item.	
No	1054	[2.3]	The input item 'transfer' may contain the grammar element. Grammar elements specify allowable spoken and character input for this input item.	
Yes	1057	[2.3]	The input item 'object' may contain the catch element, which is in effect for this input item.	
Yes	1058	[2.3]	The input item 'transfer' may contain the catch element, which is in effect for this input item.	
No	1141	[2.3.7.2.2]	Upon termination of a bridged transfer due to the caller matching an active DTMF grammar ('near_end_disconnect'), application.lastresult$ is assigned to the DTMF result.	
No	1165	[2.3.7.2.2]	Upon termination of a bridged transfer due to the caller matching an active speech grammar ('near_end_disconnect'), application.lastresult$.utterance is assigned the same value as the transfer's utterance shadow.	
No	1166	[2.3.7.2.2]	If a bridged transfer is terminated due to a reason other than the caller matching an active speech or DTMF grammar ('near_end_disconnect'), application.lastresult$ is undefined.	
Yes	1172	[2.3.1.3]	When the accept attribute of an option element is set to "approximate", the user may say a subphrase of the specified PCDATA for the option to be selected.	
No	1180	[2.3.7]	A transfer element that specifies both a 'aai' and 'aaiexpr' attribute results in an 'error.badfetch' upon loading of the document containing it.	
No	1181	[2.3.7.3]	If the destination URI is malformed, a error.connection.baddestination error is thrown.	
No	1182	[2.3.7.3]	If the destination URI type is not supported, a error.unsupported.uri error is thrown.	
No	1183	[2.3.7.3]	If the platform does not support blind transfers, a error.unsupported.transfer.blind error is thrown.	
No	1184	[2.3.7.3]	If the caller has insufficient permission to perform a call transfer (e.g. not allowed to make long distance calls, not permitted to make any transfers, etc.), destination URL is malformed, a error.connection.noauthorization error is thrown.	
No	1185	[5.1.4]	The name (session.connection.protocol.name) also represents the subobject name for protocol specific information. For instance, if session.connection.protocol.name is 'q931', session.connection.protocol.q931.uui might specify the user-to-user information property of the connection.	
No	1186	[2.3.7.3]	If the platform does not support bridge transfers, a error.unsupported.transfer.bridge error is thrown.	
No	1200	[2.3.7]	(SIP version of assertion 288) A VoiceXML document can initiate a transfer to another entity using the transfer tag, such that the Interpreter remains connected to the original caller and interpretation resumes upon termination of the transfer.	
No	1201	[2.3.7]	(SIP version of assertion 295) A bridged transfer specifying a 'connecttimeout' attribute with a W3C time specification will terminate a	

			transfer attempt if the destination entity cannot be connected to within that period of time.	
No	1202	[2.3.7]	(SIP version of assertion 289) A VoiceXML document can initiate a transfer to another entity using the transfer tag, such that the Interpreter disconnects from the caller immediately upon attempting the transfer and continues execution as it would under termination of the Session.	
No	2000	[2.3.7]	During a bridged transfer, any DTMF grammars outside the transfer element are not active and cannot be matched.	
No	2001	[2.3]	The input item 'transfer' may contain the grammar element specifying allowable DTMF input for this input item.	

Required assertions

ID	Spec. Reference	Abstract
1	[1.2.5]	For each HTTP request, the interpreter identifies itself using the User-Agent header in the format "name/version".
2	[1.2.5]	The interpreter must be able to freely sequence TTS and audio output.
7	[1.2.5]	The interpreter must be able to receive speech recognition grammar data dynamically.
8	[1.2.5]	The interpreter must be able to record audio received from the user.
11	[1.3]	If a URI does not refer to a document, the current document is assumed.
12	[1.3]	If a URI does not refer to a dialog, the first dialog is assumed.
18	[1.3.1]	A menu contains choices that can define transitions when matched.
19	[1.3.1]	A subdialog invokes a new dialog that once done, returns to the original context.
20	[1.3.1]	Upon return to the calling context all variable instances, grammars, and state information are restored.
24	[1.3.3]	An application root document's variables are defined and reachable via the application scope upon the loading of a document that specifies it as the application root.
25	[1.3.3]	An application root document's variables are not reinitialized as the user transitions between documents that both specify it as the application root.
26	[1.3.3]	An application root document's variables are no longer reachable from the application scope when the user transitions to a document not in that application.
27	[1.3.3]	An application root document's grammars are active during each time the interpreter listens, until it is unloaded.
28	[1.3.4]	A dialog can receive input matching a single grammar such that no other grammars are active.
29	[1.3.4]	A dialog can receive input matching one of several active grammars.
30	[1.3.4]	A user can direct interpretation in scenarios where grammars outside the current dialog are active, by matching one of these grammars, at which point execution transitions to the dialog containing the matched grammar.
32	[1.3.5]	The platform can throw a semantic error upon encountering an error in VoiceXML semantics.
36	[1.3.5]	Catch elements are inherited by enclosing elements by copy.
37	[1.3.6]	A link specifies a grammar that is active whenever a user is in the scope of the link.
41	[1.5]	A document may have meta elements.
42	[1.5]	A document may have var elements.
43	[1.5]	A document may have script elements.
44	[1.5]	A document may have property elements.
45	[1.5]	A document may have catch elements.

46	[1.5]	A document may have link elements.
48	[1.5.1]	The next dialog is determined by the previous dialog.
50	[1.5.1]	The version attribute is required on the vxml tag.
52	[1.5.1]	The xml:base attribute must be allowed on the vxml tag.
53	[1.5.1]	The xml:lang attribute must be allowed on the vxml tag.
54	[1.5.1]	The application attribute must be allowed on the vxml tag.
55	[1.5.1]	The value of xml:lang must be inherited down the document hierarchy by elements which also define the xml:lang attribute and do not have an alternative value.
56	[1.5.2]	The interpreter supports having an application root document and an application leaf document.
58	[1.5.2]	When a leaf document causes a root document to be loaded, none of the dialogs in the root document are executed.
59	[1.5.2]	Application root document variables are available for use by the leaf document.
61	[1.5.2]	Common ECMAScript code can be defined in the application root and used in leaf documents.
62	[1.5.2]	Application root catch handlers are default handlers for leaf documents.
72	[1.5.2]	When transitioning between two leaf documents that both specify the same application fully resolved URI then the transition must preserve the application root document's variables for use by the second leaf document.
73	[1.5.2]	A transition from an application leaf document to its own application root document caused by a 'goto' must preserve the application root document's variables for use by the root document.
74	[1.5.2]	If a transition occurs as the result of a submit between an application leaf document and its own application root document the application root document's variables must be re-initialized.
75	[1.5.2]	If a transition occurs from an application root document to itself then it must reinitialize the application root document's variables.
76	[1.5.2]	If a transition occurs from an application root document to a different application root document it must initialize the new application root document and use the new application root document's variables.
77	[1.5.2]	When a subdialog is invoked the original document and root document must be preserved for the completion of the subdialog.
80	[1.5.2]	If a document refers to a non-existent application root document, an error.badfetch event is thrown.
82	[1.5.3]	A document may contain a subdialog element.
83	[1.5.3]	A document may contain a return element.
84	[1.5.3]	Subdialogs add a new executable context when they are invoked.
85	[1.5.3]	A subdialog can be a new dialog within the existing document.
86	[1.5.3]	A subdialog can be a new dialog within a new document.
87	[1.5.3]	A subdialog can be composed of several documents.
88	[1.5.3]	A subdialog's new context may itself invoke a subdialog.
89	[1.5.4]	A VoiceXML interpreter may continue executing even after it no longer has a connection to the user.
93	[2.1]	A form may contain form items, which are subdivided into input items (<field>, <record>, <transfer>, <object>, <subdialog>) and control items (<block> and <initial>).
96	[2.1]	A form may contain <var> elements.
97	[2.1]	A form may contain event handlers: <catch>, <error>, <help>, <noinput>, and <nomatch>.
98	[2.1]	A form may contain <filled> elements.
99	[2.1]	A form may specify an id attribute.
100	[2.1]	A form may specify a scope attribute which specifies the default scope of the form's grammars.
111	[2.1.6.2.3]	FIA ends when it encounters a <goto>.
112	[2.1.6.2.3]	FIA ends when it encounters a <submit>.
113	[2.1.6.2.3]	FIA ends when it encounters an <exit>.
114	[2.1.6.2.3]	FIA ends when it encounters a <return>.

138	[2.1.6.1]	The value of the name attribute of a form item defines a dialog-scoped variable that holds the value of the form item, i.e., dialog.name is a reference to the form item variable.
143	[2.1.6.1]	A form item variable does not need to have a name attribute.
145	[2.1.5]	A form may have one or more initial form items.
147	[2.1.5]	If a form has a form level grammar, its input items can be filled in any order.
148	[2.1.5]	If a form has a form level grammar, more than one input item can be filled as a result of any one user utterance.
149	[2.1.5]	A form-level grammar can fill the following form items: <field>, <record>, <transfer>, <object> and <subdialog>.
152	[2.1.6.2.1]	For all types of form item, if the form item is assigned a value, that form item is not eligible to be visited by the FIA (unless/until it is subsequently set to undefined).
153	[2.1.6.2.1]	Using the clear tag on a form item variable will make it eligible to be visited by the FIA (provided that it does not have a cond attribute evaluating to false).
154	[2.1.6.2.1]	Using the goto nextitem will force the FIA to immediately transition to the chosen form item.
163	[2.1.6.2.1]	If the last main FIA loop resulted in a goto nextitem or goto expritem then the specified form item is selected.
165	[2.1.6.2.1]	If the last main FIA loop did not result in a goto nextitem and there is no form item which is eligible to be visited then an implicit exit is generated.
183	[2.1.6.2.3]	If in the collect phase an event occurs the appropriate catch element is identified and executed in the process phase.
185	[2.1.6.2.3]	If an event handler, executed after an event is thrown while processing a form item, transfers control with a <goto> or <submit>, the FIA resumes in the new form at the initialisation phase.
186	[2.1.6.2.3]	If an event handler, executed after an event is thrown while processing a form item, does not transfer control with a <goto> or <submit> the FIA resumes in the current form at the selection phase.
187	[2.1.6.2.3]	If an input from the collect phase matches a link than the associated link's transition is executed if present.
188	[2.1.6.2.3]	If an input from the collect phase matches a link than the associated link's event is thrown if present.
189	[2.1.6.2.3]	If a link throws an event the event is processed in the context of the current form item.
190	[2.1.6.2.3]	If an input matches a grammar in a form other than the current form, then the FIA terminates and the other form is initialized.
192	[2.1.6.2.3]	If an input matches a grammar in a form other than the current form then that form's FIA starts with this input in its process phase.
195	[2.1.6.2.3]	If an input matches a grammar in the form then each identified filled action is executed in document order.
197	[2.1.6.2.3]	While executing a filled, processing of filled actions continues after a <reprompt>.
198	[2.1.6.2.3]	If an event is thrown during the execution of a <filled>, event handler selection starts in the scope of the <filled>, which could be a form item or the form itself, and then proceeds outward by enclosing dialog scopes.
200	[2.2]	If a **menu** is given an *id* attribute, the **menu** can be referenced using **goto**.
201	[2.2]	If a menu's scope attribute is set to "dialog", the menu's grammars are active only when the user transitions into the menu.
202	[2.2]	If a menu's scope attribute is set to "document", the menu's grammars are active over the whole document (or if the menu is in the application root document, any loaded document in the application).
203	[2.2]	If a menu's dtmf attribute is set to "true", the first nine choices that have not explicitly specified a value for the dtmf attribute are given the implicit ones "1", "2", etc. Remaining choices that have not explicitly specified a value for the dtmf attribute will not be assigned DTMF values (and thus cannot be matched via a DTMF keypress).
204	[2.2]	The dtmf attribute of choice can specify a sequence of DTMF digits. Whitespace is ignored.
205	[2.2]	When the DTMF associated with the choice is matched, the appropriate action is taken based on the next, expr, event or eventexpr attribute.
206	[2.2]	When the accept attribute of choice is set to "exact", or is not set and the accept attribute of its enclosing menu is set to "exact", or neither choice's accept attribute nor menu's is set, then the text of the choice element defines the exact phrase to be recognized. The user must say the entire phrase in the same order

		in which it occurs in the choice element phrase for matching this element.
208	[2.2]	Exactly one of the next, expr, event, and eventexpr attributes must be specified. Otherwise an error.badfetch event is thrown.
209	[2.2]	When the grammar associated with a choice element is matched, the URI associated with the "next" or "expr" element is fetched and transitioned to.
210	[2.2]	When the grammar associated with a choice element is matched, the event associated with the event or eventexpr attribute is thrown.
211	[2.2]	When "next" or the result of "expr" is not a correct URI, error.badfetch is thrown.
212	[2.2]	The message attribute defines a string that is available as the variable _message inside a catch element that catches the event being thrown.
213	[2.2]	The messageexpr attribute is an ECMAScript expression evaluating to the variable _message inside the catch element which catches the event being thrown.
214	[2.2.2]	Exactly one of message or messageexpr may be specified; otherwise, an error.badfetch event is thrown.
215	[2.2]	If a grammar element is specified in choice, then the external grammar is used instead of an automatically generated grammar.
216	[2.2]	The enumerate element without content inside a prompt lists all the choices, following the order in which they appear in the menu.
217	[2.2]	The enumerate element with content defines a template specifier that will list all the choices. Two special variables are defined
218	[2.2]	An enumerate element can be used inside prompts associated with a menu element.
219	[2.2]	An enumerate element can be used inside catch element elements associated with a menu element.
220	[2.2]	An enumerate element can be used inside prompts associated with a field element that contain option elements.
221	[2.2]	An enumerate element can be used inside catch element elements associated with a field element that contain option elements.
222	[2.2]	If an enumerate element is used elsewhere, an error.semantic event is thrown.
223	[2.2]	Grammar matches within menu will update the application.lastresult$ array.
224	[2.2]	If the event handler called after matching a choice with an event or eventexpr attribute does not cause the interpreter to exit or transition control, then the FIA will clear the form item variable of the menu's anonymous field, causing the menu to be executed again.
232	[2.3.1]	The variable associated with the name attribute of the field holds the recognition result.
233	[2.3.1]	The named variable associated with the field must be unique among form items in the form; otherwise, error.badfetch is thrown.
234	[2.3.1]	If specified, the value of the expr attribute is evaluated and serves as the form item variable's initial value.
235	[2.3.1]	The default value of the expr attribute is ECMAScript undefined.
236	[2.3.1]	If the form item is initialized to a value via evaluation of the expr attribute, the form item will not be visited unless the form item variable is cleared.
237	[2.3.1]	If the cond attribute is present and its value evaluates to boolean false, the field is not visited.
238	[2.3.1]	The form item can also be visited if the cond attribute is not specified.
240	[2.3.1]	The value of the slot attribute defines the name of the grammar slot used to populate the form item variable of the field.
241	[2.3.1]	If the slot attribute is absent, the interpreter uses the value associated with the name attribute of the field to map the grammar slot to the form item variable.
242	[2.3.1]	If the value of the modal attribute is false, all active grammars are turned on while collecting this field.
243	[2.3.1]	If the value of the modal attribute is true, then only the field's grammars are enabled, and all other grammars are temporarily disabled.
244	[2.3.1]	The field element exposes a shadow variables named confidence, utterance, inputmode, and interpretation. The values of the utterance and inputmode shadow variables correspond to the values of the corresponding properties of the first object in the application-scoped lastresult$ array.

245	[2.3.1]	The confidence shadow variable is the confidence level for the name field of this interpretation and may range from 0.0-1.0. A value of 0.0 indicates minimum confidence, and a value of 1.0 indicates maximum confidence.
246	[2.3.1.1]	A field may contain a grammar element whose src attribute can specify an absolute or relative URI.
247	[2.3.1.1]	A grammar can be specified inline.
248	[2.3.1.1]	error.badfetch is thrown if a grammar specifies a src attribute and an inline grammar.
254	[2.3.1.3]	The PCDATA contained by an option element within a field generates a speech grammar.
255	[2.3.1.3]	When an option is chosen, the value attribute determines the interpretation value for the field's shadow variable and for application.lastresult$.
256	[2.3.1.3]	The dtmf attribute of an option element defines the DTMF sequence required for the option to be selected.
257	[2.3.1.3]	When the accept attribute of an option element is set to "exact", the PCDATA of the option defines the exact phrase to be recognized for the option to be selected.
258	[2.3.1.3]	Both option and grammar elements may be specified within a field.
259	[2.3.2]	The interpreter executes content contained in the block.
260	[2.3.2]	The interpreter visits the block when the cond attribute evaluates to true and the form item variable associated with the block is undefined.
261	[2.3.2]	The interpreter ignores the block when the form item variable associated with the block is defined via expr.
262	[2.3.2]	The interpreter ignores the block when the form item variable associated with the block is set via an assign.
263	[2.3.4]	The interpreter executes the subdialog associated with the src or srcexpr attribute.
267	[2.3.4]	If the called subdialog is in a separate document then variables from the calling document, and dialog scope are inaccessible to the subdialog.
268	[2.3.4]	The variables passed to the subdialog via the <param> element are accessible as variables within the dialog scope of the invoked subdialog. A <param> overrides the corresponding <var> expr attribute which is ignored.
269	[2.3.4]	The interpreter throws error.badfetch after the specified fetchtimeout when the URL associated with the src or srcexpr attribute points to a non-existent resource.
270	[2.3.4]	Exactly one of "src" or "srcexpr" must be specified; otherwise, an error.badfetch event is thrown.
271	[2.3.4]	If the subdialog returns a namelist, the filled element contained by the subdialog is executed upon return from the subdialog.
272	[2.3.5]	If an object element refers to an unknown object, the error.unsupported.objectname event is thrown.
273	[2.3.6]	Any DTMF keypress matching an active grammar terminates recording.
274	[2.3.6]	DTMF keypresses not matching an active grammar are ignored (and therefore do not terminate or otherwise affect recording).
275	[2.3.6]	If the termination grammar matched is a local DTMF grammar, the recording is placed in the record variable.
277	[2.3.6]	The variable associated with the name attribute references the recorded audio, and the audio can be played back using audio expr.
278	[2.3.6]	The variable associated with the name attribute of the record element can be submitted via the namelist of the submit element. The submitted audio is valid.
279	[2.3.6]	If a recording is created, the name$.duration shadow variable holds the duration (positive integer) of the recording in milliseconds.
281	[2.3.6]	The name$.size variable holds the size (positive integer) of the recording in bytes.
282	[2.3.6]	If the dtmfterm attribute is true, and the user terminates the recording by pressing a DTMF key which doesn't match any active DTMF grammar, then the name$.termchar shadow variable is set to the key pressed.
283	[2.3.6]	If the dtmfterm attribute is false, and user presses a key which does not match any active DTMF grammar, then recording is not terminated and the name$.termchar shadow variable is undefined when the recording terminates
284	[2.3.6]	name$.maxtime shadow variable is true if the recording was terminated because the duration specified by the maxtime attribute was reached.
312	[2.4]	When <filled> is a child of a <form>, and the mode attribute is set to "any", the filled is executed when any of

		the form items specified in the namelist has been filled.
313	[2.4]	When <filled> is a child of a <form>, and the mode attribute is set to "all", the filled is executed when all of the form items specified in the namelist have been filled.
314	[2.4]	A <filled> element within an input item cannot specify a mode.
315	[2.4]	A <filled> element within an input item cannot specify a namelist.
316	[2.4]	Control items may not be specified in the namelist of the <filled> element.
317	[2.5]	A link element may have one or more grammars that are scoped to the element containing the link.
319	[2.5]	When a link specifying next or expr is matched, the interpreter transitions to the document or dialog specified by next or expr.
320	[2.5]	A link can be a child of vxml, form, or of the form items field and initial.
321	[2.5]	A link at the form level has grammars active while the user is in that form.
322	[2.5]	If an application root document has a document-level link, its grammars are active no matter what document of the application is being executed.
323	[2.5]	A link at the vxml level has grammars that are active throughout the document.
324	[2.5]	If execution is in a modal form item, then link grammars at the form or document level are not active. [Not application?]
325	[2.5]	When a link that specifies event or eventexpr is matched, the specified event is thrown.
326	[2.5]	Events thrown as a result of matching a link are thrown at the current location in the execution, not at the location where the link is specified.
327	[2.5]	When a link is matched, application.lastresult$ is assigned.
328	[2.5]	A link may specify a message or messageexpr attribute providing additional context about the event being thrown. The message is available as the value of the _message variable within the scope of the catch element the interpreter selects to handle the event.
329	[2.5]	A link may specify a dtmf attribute that identifies the sequence of DTMF digits that cause the interpreter to activate the link.
330	[2.5]	Exactly one of next, expr, event or eventexpr must be specified; otherwise, an error.badfetch event is thrown.
331	[2.5]	Exactly one of message or messageexpr may be specified; otherwise, an error.badfetch event is thrown.
332	[3.1.1]	During a dialog, the interpreter can receive input from the user via the user's spoken utterance, such that the spoken utterance is one of those described by an active grammar.
337	[3.1.1.2]	A grammar element can specify a 'src' attribute specifying a URI which returns the data of a grammar.
338	[3.1.1.2]	A document containing a grammar element specifying both a 'src' attribute and an inline grammar description causes the Interpreter to throw an 'error.badfetch' event upon fetching of the document.
344	[3.1.2]	During a dialog, the Interpreter can receive input from the user via a set of dual tone multi frequency (DTMF) key presses, such that the sequence of presses is one of those described by an active grammar.
345	[3.1.2]	A DTMF grammar can be specified using the XML form of the SRGS grammar format, as described in the SRGS specification.
346	[3.1.2]	A "xml:lang" attribute can be specified on a DTMF grammar, but has no effect upon the grammar handling.
348	[3.1.3]	A grammar element specifying a "scope" attribute as a child of a field item throws an "error.badfetch" upon parsing. [This tests scope="dialog".]
357	[3.1.3]	A grammar element specifying a "scope" attribute as a child of a link element throws an "error.badfetch" upon parsing.
359	[3.1.3]	A grammar element specifying a "scope" attribute as a child of a menu element throws an "error.badfetch" upon parsing.
361	[3.1.6]	If a grammar matches but does not return a semantic interpretation, the raw text string for the utterance will be used.
362	[3.1.6.1]	If a form-level grammar matches and returns a semantic interpretation for a field name or slot that is a non-scalar ECMAScript variable, the field will be filled with that non-scalar ECMAScript value. The selected property may be a compound object.

363	[3.1.6.1]	The slot attribute of the field element can be used to select a sub-property of the result.
364	[3.1.6.1]	A specific slot value can fill more than one field if the slot names of the fields are the same.
365	[3.1.6.2]	If the result from a field-level grammar is a simple result, it is assigned to the input item variable.
366	[3.1.6.2]	If the result from a field-level grammar is a structure and the slot name matches a property, this property is assigned to the input item variable.
367	[3.1.6.2]	If the result from a field-level grammar is neither a simple result nor a structure with a property that matches the slot name, the entire semantic result is assigned to the input item variable.
369	[4.1.3]	The interpreter fetches and plays the URI associated with the src attribute of the audio element.
370	[4.1.3]	If src and expr are specified on an audio element, error.badfetch is thrown.
371	[4.1.3]	The interpreter evaluates, fetches, and plays the URI associated with the expr attribute of the audio element.
372	[4.1.3]	If expr is set to ECMAScript undefined the audio element is ignored.
373	[4.1.3]	If the URI associated with src is unavailable, the interpreter renders the content contained by the audio element.
374	[4.1.3]	If the URI associated with expr is unavailable, the interpreter renders the content contained by the audio element.
375	[4.1.3]	If the URI associated with src is unavailable, and the audio element doesn't contain content, no error is thrown.
376	[4.1.3]	If the URI associated with expr is unavailable, and the audio element doesn't contain content, no error is thrown.
377	[4.1.4]	If the expr attribute specifies a valid ECMAScript expression, the value element evaluates it correctly.
378	[4.1.4]	If the expr attribute specifies an invalid ECMAScript expression, error.semantic is thrown.
381	[4.1.5]	When the bargein attribute is set to false on a prompt, any DTMF input buffered in a transition state is deleted from the buffer
388	[4.1.7]	If the interval specified by the timeout attribute of the prompt element is exceeded, the platform will throw a noinput event. The default is the value specified by the timeout property
389	[4.1.7]	If several prompts are queued before a field input, the timeout of the last prompt is used.
390	[5.1]	VoiceXML variables and ECMAScript variables are contained in the same variable space.
391	[5.1]	VoiceXML variables can be used in a **script**. Variables defined in a **script** can be used in VoiceXML.
392	[5.1]	**script** can appear everywhere that **var** can appear.
393	[5.1]	Variable names that violate ECMAScript rules cause an error.semantic event to be thrown.
394	[5.1.1]	Variables declared without an explicit initial value are initialized to ECMAScript undefined.
395	[5.1.1]	Variables must be declared prior to use. Assigning to an undeclared variable does not automatically create it. Instead, it results in error.semantic being thrown.
396	[5.1.1]	In a form, variables declared by **var** and by form items are initialized every time the form is entered. These initializations take place in document order.
397	[5.1.2]	Variables can be declared in application, document, dialog and anonymous scopes. Variables declared at one scope are visible at that scope and all more local scopes.
398	[5.1.2]	Variables in session scope can be read but not written by VoiceXML documents.
399	[5.1.2]	**var** and **script** elements that are children of the application root document's **vxml** element create their variables at application scope. They are no longer accessible when another application is entered.
400	[5.1.2]	**var** and **script** elements that are children of the document's **vxml** element create their variables at document scope. They are no longer accessible when another document is entered.
401	[5.1.2]	var and script elements that are children of a form element (but not in an anonymous scope) create their variables at dialog scope. They are no longer accessible when another dialog is entered.
402	[5.1.2]	**var** and **script** elements that are children of **block**, **filled** and **catch** elements (including synonyms for **catch**, such as **nomatch**) create their variables at anonymous scope.
403	[5.1.2]	Each **block**, **filled** and **catch** element has its own new and separate anonymous scope. The scope is no longer accessible once the element is exited.

404	[5.1.2]	Scopes are not cleared when they become inaccessible. Instead, the old scope object is left to exist (or to be garbage collected) and a new one is created and linked into the scope hierarchy. References to previously-existing scope objects will continue to access the old scope objects.
405	[5.1.2]	Each scope contains a predefined variable whose name is the same as the scope that refers to the scope itself.
406	[5.1.2]	"session", "application", "document", and "dialog" are not reserved words.
407	[5.1.2]	When executing in a document that does not have a separate application root document, the application and document scopes are the same; that is, a single scope has variables named both "application" and "document" that are references to the scope itself. This includes execution in an application root document's **var** and **script** elements.
408	[5.1.3]	Variable references match the closest enclosing scope.
416	[5.1.5]	application.lastresult$ contains an array of elements, or ECMAScript undefined.
418	[5.1.5]	Each element of application.lastresult$ contains "confidence", "utterance", "inputmode" and "interpretation" properties.
419	[5.1.5]	The "confidence" property of an element of application.lastresult$ will be a number, not less than 0.0 and not greater than 1.0.
420	[5.1.5]	The "utterance" property of an element of application.lastresult$ will be a string.
421	[5.1.5]	The "inputmode" property of an element of application.lastresult$ will be either "dtmf" or "voice".
422	[5.1.5]	The "interpretation" property of an element of application.lastresult$ will contain the interpretation as described in section 3.1.5.
423	[5.1.5]	The elements of application.lastresult$ will be sorted from highest confidence score to lowest, with ties resolved by sorting by the precedence relationship among the grammars producing the interpretations.
424	[5.1.5]	Different elements in application.lastresult$ will always differ in their utterance, interpretation, or both.
425	[5.1.5]	The number of elements in application.lastresult$ is never more than the value of the property "maxnbest", and never less than one, unless the value of application.lastresult$ itself is undefined.
426	[5.1.5]	If the value of application.lastresult$ is not undefined, application.lastresult$ itself will contain the properties confidence, utterance, inputmode and intepretation corresponding to those of application.lastresult$[0].
428	[5.1.5]	application.lastresult$ is not changed after a noinput.
430	[5.2]	Catch elements are inherited as if by copy. This affects scope resolution, further thrown events, and relative URL references.
431	[5.2.1]	The interpreter throws the named event.
432	[5.2.1]	The interpreter observes event counting when an application-defined event is thrown.
433	[5.2.1]	If neither event nor eventexpr are specified then an error.badfetch event is thrown.
434	[5.2.1]	If both event and eventexpr are specified then an error.badfetch event is thrown.
435	[5.2.1]	If both message and messageexpr are specified then an error.badfetch event is thrown.
436	[5.2.2]	A VoiceXML interpreter must execute the content within the selected event handler.
437	[5.2.2]	Anonymous scope variables _event and _message are available within the event handler.
438	[5.2.2]	Assuming no cond or count attribute specified, given a field level event handler, the interpreter must select it when the corresponding event is thrown.
439	[5.2.2]	Assuming no cond or count attribute specified and no field-level event handler, given a form-level event handler, the interpreter must select it when the corresponding event is thrown.
440	[5.2.2]	Assuming no cond or count attribute specified and no field- or form-level event handler, given a document-level event handler, the interpreter must select it when the corresponding event is thrown.
441	[5.2.2]	Assuming no cond or count attribute specified and no field-, form-, or document-level event handler, given an application-level event handler, the interpreter must select it when the corresponding event is thrown.
442	[5.2.2]	If the cond attribute of the most inner-scoped event handler evaluates to false, the interpreter must not select it and instead select the next qualifying event handler.
443	[5.2.2]	The cond attribute of an event handler must be evaluated in the context of the scope where the event is thrown.

444	[5.2.2]	The interpreter executes the event handler with highest count attribute not greater than the actual event count.
446	[5.2.2]	Event count matching takes precedence over scoping, ie on the second occurrence of an event a root document level handler with count=2 takes precedence over a field level handler with count=1
448	[5.2.2]	The interpreter maintains a separate count for each event when multiple events specified in the event attribute of the catch element.
449	[5.2.2]	If unspecified, the count attribute of the catch element defaults to 1.
450	[5.2.2]	The interpreter selects the first handler in document order when two or more event handlers have the same count attribute and scope.
451	[5.2.2]	The interpreter executes a handler that speciifies a space-delimited list of events in the event attribute when the corresponding events are thrown.
452	[5.2.2]	An event handler catches all events if the event attribute not specified.
453	[5.2.2]	An event handler catches all events for which the event attribute specifies a token prefix of the event (NB token prefix matching is done after removing trailing dots.)
454	[5.2.2]	An event handler does not catch events for which the event attribute specifies a string prefix of the event which is not a token prefix.
455	[5.2.3]	Exactly one of "next", "expr", "event" or "eventexpr" must be specified; otherwise, an error.badfetch event is thrown. Exactly one of "message" or "messageexpr" may be specified; otherwise, an error.badfetch event is thrown.
457	[5.2.3]	When event or eventexpr is specified, and the link is matched, the corresponding event is thrown.
458	[5.2.3]	When next or expr is specified, and the link is matched, the interpreter navigates to the specified URI.
460	[5.2.3]	The noinput element is executed when an noinput event is thrown.
461	[5.2.3]	The noinput element obeys the same selection and execution rules as a catch element whose event attribute is set to "noinput".
462	[5.2.3]	The noinput element has exactly equal precedence to a catch element that specifies the noinput event, ie the choice between a catch element and a noinput element at the same scope level and count is made on the basis of document order.
464	[5.2.3]	The nomatch element is executed when a nomatch event is thrown.
465	[5.2.3]	The nomatch element obeys the same selection and execution rules as a catch element whose event attribute is set to "nomatch".
466	[5.2.3]	The nomatch element has exactly equal precedence to a catch element that specifies the nomatch event, ie the choice between a catch element and a nomatch element at the same scope level and count is made on the basis of document order.
468	[5.2.3]	The error element is executed when an error event is thrown.
469	[5.2.3]	The error element is executed when a more specific error event is thrown (e.g. error.badfetch)
470	[5.2.4]	The error element is executed when an application-defined event prefixed with "error" is thrown.
471	[5.2.3]	The error element has exactly equal precedence to a catch element that specifies the error event, ie the choice between a catch element and an error element at the same scope level and count is made on the basis of document order.
472	[5.2.3]	The error element obeys the same selection rules as a catch element whose event attribute is set to "error".
473	[5.2.3]	The error element obeys the same execution rules as a catch element whose event attribute is set to "error".
474	[5.2.3]	The help element is executed when an help event is thrown.
475	[5.2.3]	The help element obeys the same selection and execution rules as a catch element whose event attribute is set to "help".
476	[5.2.3]	The help element has exactly equal precedence to a catch element that specifies the help event, ie the choice between a catch element and a help element at the same scope level and count is made on the basis of document order.
478	[5.2.4]	An event handler catches all events if the event attribute consists of a single dot (".").
479	[5.2.4]	Event handlers which specify more specific event names, eg event.foo.bar have equal precedence with event handlers that specify less specific event names, eg event.foo

480	[5.2.4]	Event handlers are executed in the context in which the event was thrown. Check for the existence of variables in the scope in which the event was thrown.
481	[5.2.4]	Event handlers are executed in the context in which the event was thrown. Check that URLs (audio/@src, goto/@next, script/@src, etc) are resolved relative to the document in which the event was thrown.
482	[5.2.5]	The interpreter must support a default cancel event handler. No audio is provided, the interpreter does not reprompt and does not exit.
483	[5.2.5]	The interpreter must support a default exit event handler. No audio is provided and the interpreter exits.
484	[5.2.5]	The interpreter must support a default error event handler. The interpreter provides platform-specific audio and exits.
485	[5.2.5]	The interpreter must support a default help event handler. The interpreter provides platform-specific audio and reprompts.
486	[5.2.5]	The interpreter must support a default noinput event handler. No audio is provided and the interpreter reprompts.
487	[5.2.5]	The interpreter must support a default nomatch event handler. The interpreter provides platform-specific audio and reprompts.
488	[5.2.5]	The interpreter must support a default maxspeechtimeout event handler. The interpreter provides platform-specific audio and reprompts.
489	[5.2.5]	The interpreter must support a default connection.disconnect event handler. No audio is provided and the interpreter exits.
490	[5.2.5]	For any other unhandled event, the interpreter provides platform-specific audio and exits
494	[5.2.6]	When the user hangs up the connection.disconnect.hangup event is thrown.
497	[5.2.6]	The nomatch event is thrown if the user says something and is not recognized.
498	[5.2.6]	The maxspeechtimeout event is thrown if the user's input exceeds the maxspeechtimeout property.
499	[5.2.6]	If the interpreter encounters a syntax error while loading a document (eg no vxml element), it throws error.badfetch in the requesting document.
500	[5.2.6]	If an invalid URL is specified for a document fetch, the interpreter throws error.badfetch.http.404 in the requesting document.
501	[5.2.6]	If a document fetch times out, the interpreter throws error.badfetch in the requesting document.
502	[5.2.6]	If a server returns HTTP error code 4xx the event error.badfetch.http.4xx should be thrown in the requesting document.
503	[5.2.6]	If a server returns HTTP error code 5xx the event error.badfetch.http.5xx should be thrown in the requesting document
505	[5.2.6]	If an unsupported language is specified for speech recognition, the interpreter throws error.unsupported.language.
506	[5.2.6]	If an unsupported language is specified for speech synthesis, the interpreter throws error.unsupported.language.
508	[5.2.6]	If an unsupported grammar format is specified, the interpreter throws error.unsupported.format.
509	[5.3]	If an executable element generates an error, the error is thrown immediately. Subsequent executable elements in that block of procedural logic are not executed.
510	[5.3.1]	A variable declared at document scope is accessible within an anonymous scope contained within the same document.
511	[5.3.1]	Declaring a variable at an anonymous scope, the variable is not accessible within another anonymous scope.
512	[5.3.1]	A variable declared at a higher scope (e.g. document) is shadowed by a variable at a lower scope (e.g. anonymous).
513	[5.3.1]	When declaring the same variable multiple times with different initial values in the same scope, declarations will apply in document order.
514	[5.3.2]	When expr is set to a valid expression in an assign, the named variable is set correctly.
515	[5.3.2]	If name is set to an undefined variable in an assign, error.semantic is thrown.

516	[5.3.2]	If expr contains an undefined variable in an assign, error.semantic is thrown.
517	[5.3.2]	If expr contains a semantic error (e.g.it contains a non-existent function named x), an error.semantic is thrown.
518	[5.3.3]	When the namelist attribute of the clear element specifies a specific set of one or more form item variables, only those form items are cleared.
519	[5.3.3]	When the namelist attribute of the clear element is omitted, all form items in the current form are cleared.
520	[5.3.3]	If variables that are not form item variables are specified in the "namelist" attribute of the **clear** element, those variables are set to ECMAScript undefined.
521	[5.3.4]	if parent cond of if element evaluates to false, the interpreter executes the content following the else element up to the closing if tag.
522	[5.3.4]	if parent cond of if element evaluates to true, the interpreter does not execute the content following the else element up to the closing if tag.
523	[5.3.5]	If a **prompt** element appears in executable content, the "count" attribute is ignored, but the "cond" element is respected.
524	[5.3.5]	Wherever **prompt** is allowed, PCDATA is treated as if it had been wrapped in **prompt /prompt**.
525	[5.3.6]	Execution of a **reprompt** element causes the FIA to perform normal selection and queueing of prompts after execution of a catch element.
526	[5.3.7]	Setting next to a fully-qualified URL (excluding fragment identifier) that points to an existing VoiceXML document causes the INTERPRETER to transition to that document and begin execution of the first form.
527	[5.3.7]	Setting next to a relative URL (excluding fragment identifier) that points to an existing VoiceXML document causes the INTERPRETER to transition to that document and begin execution of the first form.
528	[5.3.7]	Setting next to a fully-qualified URL including fragment identifier that points to an existing VoiceXML document and form causes the INTERPRETER to transition to the document and begin execution of the specified form.
529	[5.3.7]	Setting next to a relative URL including fragment identifier that points to an existing VoiceXML document and form causes the INTERPRETER to transition to that document and begin execution of the specified form.
530	[5.3.7]	Setting expr to an ECMAScript expression that evaluates to a fully-qualified URL (excluding fragment identifier) that points to an existing VoiceXML document causes the INTERPRETER to transition to that document and begin execution of the first form.
531	[5.3.7]	Setting next to a URL that points to a non-existent VoiceXML document causes the interpreter to throw a catchable error.badfetch event.
532	[5.3.7]	Setting expr to a URL that points to a non-existent VoiceXML document causes the INTERPRETER to thrown a catchable error.badfetch event.
533	[5.3.7]	Setting next to a fragment identifier that identifies an existing form in the current document causes the INTERPRETER to transition to that form with the state of the current document and application in tact.
534	[5.3.7]	Setting next to a non-fragment URL of the current document causes the INTERPRETER to transition to the first form of the current document and reset the state of the document including any variables.
535	[5.3.8]	The URI referenced by submit's next or expr attribute is always fetched, even if it is just a fragment.
536	[5.3.8]	If submit has a namelist attribute, all and only those variables are submitted.
537	[5.3.8]	If submit has no namelist attribute, all and only the named input items in the current form are submitted.
538	[5.3.8]	Both declared VoiceXML variables and declared ECMAScript variables can be submitted. This includes properties of ECMAScript objects.
539	[5.3.8]	Interpreters must support GET and POST as submit methods. GET must be the default.
541	[5.3.8]	When an ECMAScript variable is submitted, its value is first converted to a string.
542	[5.3.8]	Specifying a URL that points to a non-existent resources causes the INTERPRETER to throw a catchable error.badfetch event.
543	[5.3.8]	Exactly one of "next" or "expr" must be specified; otherwise, an error.badfetch event is thrown.
544	[5.3.9]	Executing exit must terminate all loaded documents and return control to the interpreter context.
545	[5.3.9]	If both "expr" and "namelist" attributes of exit are specified, an error.badfetch event is thrown.

546	[5.3.10]	When **return** is executed while not inside a subdialog context, an error.semantic event is thrown.
547	[5.3.10]	Exactly one of "event", "eventexpr" or "namelist" may be specified; otherwise, an error.badfetch event is thrown.
549	[5.3.10]	The interpreter throws throws error.semantic if a return element is encountered when not executing in the context of a subdialog.
550	[5.3.10]	If the namelist attribute is specified, the specified variables become properties of an ECMAScript object accessible via the name attribute of the calling subdialog element.
551	[5.3.10]	If the event or eventexpr attribute is specified, the named event is thrown at the invocation point.
552	[5.3.11]	When the interpreter executes a disconnect element, it must drop the call.
553	[5.3.11]	When the interpreter executes a disconnect element, it must throw a catchable connection.disconnect.hangup event.
554	[5.3.11]	When the caller hangs up, the interpreter must throw a catchable connection.disconnect.hangup event.
555	[5.3.13]	ECMAScript expressions within the PCDATA in log must be evaluated in document order.
556	[5.3.12]	A script element is executed in the scope of its containing element.
557	[5.3.12]	When a script element declared in a block element contains a function, an attempt to call that function from another block causes the INTERPRETER to throw an error (error.semantic?)
558	[5.3.12]	A variable declared using the var element is accessible to a script declared at equal more local scope.
559	[5.3.12]	A variable declared within an inline or externally referenced script block is accessible from a var or assign element declared at equal or more local scope.
560	[5.3.12]	When the script element specifies a src attribute that references a URL that references a non-existent resource, the interpreter throws error.badfetch.
561	[5.3.12]	Either an "src" attribute or an inline script (but not both) must be specified; otherwise, an error.badfetch event is thrown.
562	[5.3.13]	The use of the log element has no side-effects on interpretation.
563	[6.1.1]	The interpreter context is always required to honor the safe fetchhint.
564	[6.1.1]	When transitioning from one dialog to another if the referenced URI names a document (e.g. "doc.vxml#dialog"), or if query data is provided (through POST or GET), then the new document goes through its initialization phase . Applies to **subdialog**, **goto**, **submit**, **link**, or **choice** element.
565	[6.1.1]	If a URI reference in a **goto** transition contains only a fragment (e.g., "#my_dialog"), then no document is fetched, and no initialization of that document is performed.
566	[6.1.1]	If a URI reference in a **submit** transition is accompanied by a query string or by a namelist attribute there will a fetch and the new document is initialized.
567	[6.1.1]	If the URI reference to the root document contains a query string or a namelist attribute, the root document is fetched.
568	[6.1.1]	If specified, fetchaudio plays during a long fetch.
569	[6.1.1]	If fetchaudio is not specified, but a non-empty fetchaudio property exists, fetchaudio plays during a long fetch.
570	[6.1.1]	If not specified, and the fetchaudio property is not set, fetchaudio does not play during a long fetch.
571	[6.1.1]	If an error occurs retrieving fetchaudio from its URI, no badfetch event is thrown and no audio is played during the fetch.
572	[6.1.1]	If content is not returned within the specified fetchtimeout, an error.badfetch event is thrown.
573	[6.1.1]	If content is returned within the specified fetchtimeout, document processing proceeds as normal.
574	[6.1.2]	If maxage is specified and the resource age is greater than the maxage, the interpreter must revalidate against the server
576	[6.1.2]	If maxstale attribute is specified and the resource age has exceeded its expiration time by more than the maxstale time, the interpreter must revalidate against the server.
577	[6.1.2]	Setting maxage of zero forces the interpreter to do revalidate the resource against the server.
580	[6.1.3]	If the interpreter prefetches a resource, and the URI is computed with an expr attribute, the interpreter context must not move the fetch up before any assignments to the expression's variables.

585	[6.2.1]	The interpreter must successfully parse and execute a VoiceXML document containing zero or more meta elements with name and content attributes.
586	[6.2.1]	The interpreter must successfully parse and execute a VoiceXML document containing zero or more meta elements with name and http-equiv attributes.
587	[6.2.1]	Exactly one of "name" or "http-equiv" must be specified; otherwise, an error.badfetch event is thrown.
589	[6.2.2]	An interpreter MUST successfully parse and execute a VoiceXML document containing a metadata element containing any valid schema including the recommended general metadata properties defined in the Dublin Core Metadata Initiative.
591	[6.3]	A property specified at a more local scope (e.g. form) takes precedence over the same property specified at an outer scope (e.g. document).
592	[6.3.2]	When the "confidencelevel" property is specified, the interpreter rejects results (throwing a nomatch event) when the confidence level of recognizer is less than that of the property.
595	[6.3.2]	When the "speedvsaccuracy" property is specified with value M, the recognizer is at least as fast as when the specified value is N, such that M < N, given the same input and other context.
596	[6.3.2]	When the "completetimeout" property is specified, the recognizer will wait the specified number of seconds of silence before determining that the user has finished speaking a matching utterance (or will not speak).
597	[6.3.2]	When the "incompletetimeout" property is specified, the recognizer will wait the specified number of seconds of silence before determining that the user has finished speaking and the utterance does not match a grammar.
598	[6.3.3]	When the "interdigittimeout" property is specified, and the recognizer has received DTMF input but could possibly match a grammar if more digits were entered, it will wait the specified number of seconds before determining that no more digits will be entered.
599	[6.3.3]	When the "termtimeout" property is specified, and the recognizer has received DTMF input matching a grammar and the termchar is non-empty, the user can enter an optional termchar DTMF. If the user fails to enter this optional DTMF within termtimeout, the recognition ends and the recognized value is returned.
603	[6.3.4]	The value of the timeout property controls the default time after which a noinput event is thrown by the platform.
604	[6.3.5]	If any of the various *fetchhint properties is set to "safe", content of that time is never fetched until it is needed.
605	[6.3.5]	A cached resource of a certain type must be reloaded if the *maxage property for its type is less than its current age.
606	[6.3.5]	A cached resource of a certain type must be reloaded if the *maxstale property for its type is less than its current staleness.
607	[6.3.5]	Fetchaudio will not begin playing until the amount of time specified in the fetchaudiodelay property has elapsed.
608	[6.3.5]	Fetchaudio, once it begins playing, will not be interrupted until at least the amount of time specified in the fetchaudiominimum property.
611	[6.4]	Exactly one of "expr" or "value" on param must be specified; otherwise, an error.badfetch event is thrown.
612	[6.4]	When param is contained in a subdialog element, the values specified by it are used to initialize dialog var elements in the subdialog that is invoked.
613	[6.5]	Time designations follow those used in W3C's Cascading Style Sheet recommendation. The valid time unit identifiers are ms (milliseconds, the default) and s (seconds).
614	[2.3.2]	When the block cond attribute evaluates to false, the interpreter does not execute the element or its contents.
620	[5.3.8]	Variables associated with the namelist should be URI escaped.
621	[2.3.4]	When the element's cond attribute evaluates to false, the interpreter does not execute the element or its contents.
623	[4.1]	When the prompt element's *cond* attribute evaluates to false, the interpreter does not execute the element or its contents.
626	[1.2.5]	An implementation platform must support text-to-speech
627	[1.2.5]	Audio files are referred to by URI

628	[1.2.5]	The implementation must report characters (DTMF entered by the user)
629	[1.2.5]	If an audio input resource is not available, an error.noresource event must be thrown
632	[1.3.1]	Each field may specify a grammar that defines the allowable inputs for that field.
633	[1.3.5]	Handling Platform generated events
634	[1.3.5]	Handling Interpreter generated events
638	[1.5.1]	If the xml:base attribute is defined all relative references will use the defined base URI.
640	[1.5.1]	vxml- xml:lang can be omitted and a platform specific default shall be used
649	[1.5.2]	Transition to doc with application attr same (with query string)
650	[1.5.2]	Transition to doc with application attr same (with fragment)
654	[1.5.3]	Subdialog results are accessed through properties of the variable defined by the name attribute of the subdialog element.
655	[1.5.3]	If subdialog execution is transferred to another subdialog using goto, when the second dialog returns, control is returned directly to the dialog.
656	[1.5.3]	If subdialog execution calls a second subdialog execution, when the second dialog returns, control is returned directly to the calling subdialog dialog.
1001	[5.3.7]	Exactly one of "next", "expr", "nextitem" or "expritem" must be specified; otherwise, an error.badfetch event is thrown
1002	[5.3.7]	Setting expr to an ECMAScript expression that evaluates to a relative URL (excluding fragment identifier) that points to an existing VoiceXML document causes the INTERPRETER to transition to that document and begin execution of the first form.
1003	[5.3.7]	Setting expr to an ECMAScript expression that evaluates to a fully-qualified URL including fragment identifier that points to an existing VoiceXML document causes the INTERPRETER to transition to that document and begin execution of the specified form.
1004	[5.3.7]	Setting expr to an ECMAScript expression that evaluates to a relative URL including fragment identifier that points to an existing VoiceXML document causes the INTERPRETER to transition to that document and begin execution of the specified form.
1005	[5.3.7]	Setting nextitem to an existent form item of the current form causes the INTERPRETER to transition to that specified form item and continue execution.
1006	[5.3.7]	Setting expritem to an ECMAScript expression that evaluates to an existent form item of the current form causes the INTERPRETER to transition to that specified form item and continue execution.
1007	[5.3.7]	Setting nextitem to a non-existent form item (in the current form) causes the INTERPRETER to thrown a catchable error.badfetch event
1008	[5.3.7]	Setting expritem to an ECMAScript expression that evaluates to a non-existent form item (in the current form) causes the interpreter to throw a catchable error.badfetch event.
1010	[2.3.6]	If the modal attribute is set to false, and DTMF input matches an active non-local grammar, then the recording is terminated, the record variable is undefined and control is transfered to the element containing the matched grammar.
1011	[2.3.6]	If the modal attribute is set to true, DTMF input cannot match any non-local grammars.
1012	[2.3.6]	If the expr attribute evaluates to a defined value, then the record element is not visited.
1013	[2.3.6]	If the cond attribute is specified and evaluates after conversion to boolean to true, then the record element is visited.
1014	[2.3.6]	If the cond attribute is specified and evaluates after conversion to boolean to false, then the record element is not visited.
1016	[2.3.6]	Recording is terminated when user hangs up and a disconnect event is thrown. If audio has been collected, then the audio recorded up until hangup is available through the record variable.
1017	[2.3.6]	The value of the record variable 'name' is also available as dialog.'name'.
1018	[2.3.6]	If the beep attribute is set to true, then a tone is emitted prior to recording.
1019	[2.3.6]	If DTMF input is received before the end of playback of record prompts, or the end of the timeout period, then no audio is collected, the record variable remains undefined.
1022	[5.1.2]	New session variables cannot be declared by VoiceXML documents.

1025	[2.3]	The form item initial has a result variable, specified by the name attribute. This variable may be given an initial value with the expr attribute.
1026	[2.3]	The form item subdialog has a result variable, specified by the name attribute. This variable may be given an initial value with the expr attribute.
1032	[2.3]	The form item 'initial' has a guard condition specified with the cond attribute. A form item is visited if it is not filled and its cond is not specified or evaluates, after conversion to boolean, to true.
1037	[2.3]	The input item 'field' may contain the filled element. Filled elements contain an action to execute after the result input item variable is filled in.
1038	[2.3]	The input item subdialog may contain the filled element. Filled elements contain an action to execute after the result input item variable is filled in.
1040	[2.3]	The input item record may contain the filled element. Filled elements contain an action to execute after the result input item variable is filled in.
1042	[2.3]	The input item 'field' may contain the property element. Property elements specify properties that are in effect for this input item.
1043	[2.3]	The input item 'subdialog' may contain the property element. Property elements specify properties that are in effect for this input item.
1045	[2.3]	The input item record may contain the property element. Property elements specify properties that are in effect for this input item.
1047	[2.3]	The input item 'field' may contain the prompt element. Prompt elements specify prompts to play when visiting this input item.
1048	[2.3]	The input item 'subdialog' may contain the prompt element. Prompt elements specify prompts to play when visiting this input item.
1050	[2.3]	The input item 'record' may contain the prompt element. Prompt elements specify prompts to play when visiting this input item.
1053	[2.3]	The input item 'record' may contain the grammar element. Grammar elements specify allowable DTMF input for this input item.
1055	[2.3]	The input item 'field' may contain the catch element, which is in effect for this input item.
1056	[2.3]	The input item subdialog may contain the catch element, which is in effect for this input item.
1059	[2.3]	The input item 'record' may contain the catch element, which is in effect for this input item.
1061	[2.3]	An 'initial' element may contain catch elements.
1062	[2.1]	A form may contain form-level <grammar> elements.
1063	[2.1]	A form may contain <link> elements.
1064	[2.1]	A form may contain <property> elements.
1065	[2.1]	A form may contain <script> elements.
1066	[2.1.5]	A form-level grammar cannot fill <block>, <initial> or <var>.
1067	[2.1.6.1]	On entering a form, all form item variables and variables declared in a form by <var> are initialised by the expr attribute if defined.
1068	[2.1.6.1]	On entering a form, all form item variables and variables declared in a form by <var> are initialised to undefined if no expr attribute is defined.
1069	[2.1.6.1]	Form item variables and variables declared in a form by <var> are initialised in document order.
1070	[2.1.6.1]	Form level <script> elements are evaluated in document order at the same time as form item variables are initialised.
1071	[2.1.6.1]	When a form is entered, the prompt counter is initialised to 1 for every input item. Thus the prompt with count=1 will always be played first for all input items.
1072	[2.1.6.1]	When a form is entered, the prompt counter is initialised to 1 for every <initial>. Thus the prompt with count=1 will always be played first for any <initial> item.
1074	[2.1.6.2.1]	For every type of form item except <initial>, that item will be selected by the FIA if it is the first eligible form item in document order and the previous loop of the FIA did not end in a goto nextitem
1075	[2.1.6.2.1]	For every type of form item, the form item is not eligible to be visited by the FIA if the form item variable is not

		undefined. This applies even if there is a cond attribute that evaluates to true.
1076	[2.1.6.2.1]	For every type of form item, the form item is not eligible to be visited by the FIA if the form item has a cond attribute evaluating to false.
1077	[2.1.6.2.1]	For every type of form item, the form item is eligible to be visited by the FIA if the form item has a cond attribute evaluating to true and the form item variable is undefined.
1078	[2.1.6.2.1]	For every type of form item, the form item is eligible to be visited by the FIA if the form item has no cond attribute and the form item variable is undefined.
1080	[2.1.6.2.1]	For every type of form item, the cond attribute is evaluated in the dialog scope.
1081	[3.1.4]	If no grammars are active when an input is expected, the platform must throw an error.semantic event. The error will be thrown in the context of the executing element.
1082	[2.1.6.2.3]	If execution of a block throws an event, the appropriate event handler is executed.
1083	[2.1.6.2.3]	If a document scope menu grammar is matched while processing another form, control transitions to the target of the the matched <choice> element's next or expr attribute.
1084	[2.1.6.2.3]	If a document scope menu grammar is matched while processing another form, the matched <choice> element's event or eventexpr is thrown.
1085	[2.1.6.2.3]	If a document scope menu grammar is matched while processing another form and the matched <choice> element's event or eventexpr is thrown and the event handler does not transition to a new form, the FIA resumes in the <menu> after clearing the menu's anonymous form item variable.
1086	[2.1.6.2.3]	After a field level grammar is matched, a form level <filled> element will be queued for execution if the field item variable is specified in the namelist of the <filled> element and the mode is "any".
1087	[2.1.6.2.3]	After a field level grammar is matched, a form level <filled> element will be queued for execution if the field is the only entry in the namelist and the mode is "all".
1088	[2.1.6.2.3]	After a field level grammar is matched, a form level <filled> element will be queued for execution if the namelist is empty and the mode is "any".
1089	[2.1.6.2.3]	After a field level grammar is matched, a form level <filled> element will be queued for execution if the namelist is empty, the mode is "all" and the field is the only input item in the form.
1090	[2.1.6.2.3]	After a field level grammar is matched, a form level <filled> element will not be queued for execution if the field item variable is not specified in the (non-empty) namelist of the <filled> element.
1091	[2.1.6.2.3]	After a field level grammar is matched, a form level <filled> element will not be queued for execution if the namelist of the <filled> element references other input items in addition to this field and the mode is "all".
1092	[2.1.6.2.3]	After a field level grammar is matched, a form level <filled> element will not be queued for execution if the namelist is empty, the mode is "all" and the form has other input items.
1093	[2.1.6.2.3]	If a form level grammar is matched, field level <filled> elements are queued for execution if and only if the field has just been assigned a value by this input.
1094	[2.1.6.2.3]	After a form level grammar is matched, a form level <filled> element will be queued for execution if all the fields listed in the namelist of the <filled> have just been assigned a value.
1095	[2.1.6.2.3]	After a form level grammar is matched, a form level <filled> element will be queued for execution if at least one of the fields listed in the namelist of the <filled> has just been assigned a value and the mode is "any".
1096	[2.1.6.2.3]	After a form level grammar is matched, a form level <filled> element will be queued for execution if the namelist is empty, the mode is "all" and all input item variables have just been assigned a value.
1097	[2.1.6.2.3]	After a form level grammar is matched, a form level <filled> element will be queued for execution if the namelist is empty, the filled mode is "any" and at least one input item variable has just been assigned a value.
1098	[2.1.6.2.3]	After a form level grammar is matched, a form level <filled> element will not be queued for execution if the mode is "all" and at least one of the fields listed in the namelist of the <filled> has not just been assigned a value.
1099	[2.1.6.2.3]	After a form level grammar is matched, a form level <filled> element will not be queued for execution if none of the fields listed in the namelist of the <filled> has just been assigned a value and the mode is "any" .
1100	[2.1.6.2.3]	After a form level grammar is matched, a form level <filled> element will not be queued for execution if the namelist is empty, the mode is "all" and at least one input item variables has not just been assigned a value.
1101	[2.1.6.2.3]	After a form level grammar is matched, a form level <filled> element will not be queued for execution if the

		namelist is empty, the filled mode is "any" and no input item variable has just been assigned a value.
1102	[2.1.6.2.3]	If no input items are filled by a user input and no events are thrown, the FIA resumes at the next iteration of the main loop.
1103	[2.1.6.2.3]	<goto nextitem> does not reset conditions associated with items in the targeted form.
1104	[2.1.6.2.3]	<goto nextitem> does not reset form item variables in the targeted form item.
1105	[2.1.6.2.3]	<goto nextitem> will cause the target form item's prompt to be played even if it has already been visited.
1107	[2.3.3]	If one or more of a form's field item variables are set by user input, then all <initial> form item variables within the form are set to true, before any <filled> actions are executed.
1108	[2.3.3]	If an <initial> form item variable has been explicitly cleared after a previous user input, the <initial> form item variable will only be set when a new grammar match sets a field item variable.
1109	[2.3.3]	Grammars with <field> scope are not active within <initial>.
1110	[2.3.3]	Explicit assignment of values to input item variables does not affect the value of an <initial>'s form item variable.
1111	[2.3.3]	<initial> elements may contain audio prompts, properties , and event handlers.
1112	[2.3.3]	When an event is thrown while visiting an <initial> element, the <initial> element is the innermost scope for finding an appropriate event handler, and therefore events may be caught by event handlers specified within the <initial> element.
1113	[2.3.3]	<initial> elements have event counters that follow the normal event count semantics.
1114	[2.3.3]	<initial> elements collect a user input, i.e. they cause queued prompts to be played, grammars to be activated and a speech recognition process to be started.
1115	[3.1.6.3]	Test of example table: form-level result 'hello'.
1116	[3.1.6.3]	Test of example table: form-level result '{ x: valueX }'.
1117	[3.1.6.3]	Test of example table: form-level result '{ y: valueY }'.
1118	[3.1.6.3]	Test of example table: form-level result '{ z: valueZ }'.
1119	[3.1.6.3]	Test of example table: form-level result '{ x: valueX , y: valueY , z: valueZ }'.
1120	[3.1.6.3]	Test of example table: form-level result '{ a: valueA , b: value B }'.
1121	[3.1.6.3]	Test of example table: field-level X result 'hello'.
1122	[3.1.6.3]	Test of example table: field-level X result '{ x: valueX }'.
1123	[3.1.6.3]	Test of example table: field-level X result '{ y: valueY }'.
1124	[3.1.6.3]	Test of example table: field-level X result '{ z: valueZ }'.
1125	[3.1.6.3]	Test of example table: field-level X result '{ x: valueX , y: valueY , z: valueZ }'.
1126	[3.1.6.3]	Test of example table: field-level X result '{ a: valueA , b: value B }'.
1127	[3.1.6.3]	Test of example table: field-level Z result 'hello'.
1128	[3.1.6.3]	Test of example table: field-level Z result '{ x: valueX }'.
1129	[3.1.6.3]	Test of example table: field-level Z result '{ y: valueY }'.
1130	[3.1.6.3]	Test of example table: field-level Z result '{ z: valueZ }'.
1131	[3.1.6.3]	Test of example table: field-level Z result '{ x: valueX , y: valueY , z: valueZ }'.
1132	[3.1.6.3]	Test of example table: field-level Z result '{ a: valueA , b: value B }'.
1135	[4.1.3]	If neither 'src' nor 'expr' are specified on an audio element, an error.badfetch event is thrown.
1138	[2.1.6.2.1]	Assigning a form item's variable to undefined via ECMAScript makes it eligible to be visited by the FIA, without resetting the error counters associated with the item.
1139	[2.1.6.2.3]	<goto nextitem> does not reset event counters associated with items in the targeted form.
1140	[2.1.6.2.3]	<goto nextitem> does not reset prompt counters associated with items in the targeted form.
1143	[2.1.6.2.3]	While executing a filled, if a submit is encountered the remaining filled actions are skipped.
1144	[2.1.6.2.3]	While executing a filled, if a disconnect is encountered the remaining filled actions are skipped.
1145	[2.1.6.2.3]	While executing a filled, if an exit is encountered the remaining filled actions are skipped.

1146	[2.1.6.2.3]	While executing a filled, if a return is encountered the remaining filled actions are skipped.
1147	[2.1.6.2.3]	While executing a filled, if a goto is encountered the remaining filled actions are skipped.
1148	[2.1.6.2.3]	While executing a filled, if a throw is encountered the remaining filled actions are skipped.
1149	[5.3.8]	The interpreter throws "error.badfetch.http.404" if the HTTP server returns a 404 before the fetchtimeout. If, however, the fetchtimeout interval elapses before the server returns a 404, the interpreter throws "error.badfetch".
1150	[5.3.8]	If Querystring & namelist variable ,both are mentioned with method GET then both values need to be submitted
1152	[5.3.13]	The <log> element may contain any combination of text (CDATA) and <value> elements.
1156	[2.3.4]	When the subdialog returns, its execution context is deleted. All subdialog context variable bindings are lost.
1157	[2.3.4]	If a subdialog URI has a query string and the subdialog has a namelist attribute the namelist variables are additionally submitted.
1158	[2.3.4]	When there is no fragment, the subdialog invoked is the lexically first dialog in the document.
1159	[2.3.4]	It is a semantic error to attempt to set a form item variable or an undeclared variable using <param>
1161	[5.3.13]	The label attribute may be used, for example, to indicate the purpose of the log.
1162	[5.3.9]	The exit element does not throw an "exit" event.
1163	[5.3.13]	When expr is set in a log, the expression evaluated to a string is present in a logging or debug message.
1167	[6.1.1]	If fetchtimeout is not specified, but a non-empty fetchtimeout property exists, then if content is not returned within the specified fetchtimeout property, an error.badfetch event is thrown.
1168	[6.1.1]	If fetchtimeout is not specified, but a non-empty fetchtimeout property exists, then if content is returned within the specified fetchtimeout property, document processing proceeds as normal.
1169	[5.3.12]	The <script> element has the charset attributes: The character encoding of the script designated by src. UTF-8 and UTF-16 encodings of ISO/IEC 10646 must be supported (as in [XML]). The default value is UTF-8.
1170	[6.1.1]	If a fetchaudio is played during fetch of a document, the play is interrupted when the document is loaded.
1171	[2.3.1]	If the specified builtin type is not supported by the platform, an error.unsupported.builtin event is thrown.
1173	[2.2]	DTMF properties apply to recognition of the DTMF digits specified by dtmf attribute of choice.
1174	[2.5]	Any URIs in the content of a link are resolved lexically, i.e. according to the base URI (see xml:base in Section 1.5.1) for the document in which the link is defined.
1175	[2.5]	any URIs in an attribute of a link element are resolved dynamically, i.e. according to the base URI in effect when the link s grammar is matched.
1176	[2.5]	any ECMAScript expressions in an attribute of a link element are evaluated dynamically, i.e. in the scope and execution context in effect when the grammar is matched.
1179	[5.3.2]	When an ECMAScript object, e.g. "obj", has been properly initialized then its properties, for instance "obj.prop1", can be assigned without explicit declaration.

Assertions in Dispute

Note: These assertions identify inconsistencies in VoiceXML or SRGS. As such, there is no "correct" behavior. The observed behavior is indicated here for informational purposes only for the benefit of the application developer. If and when these issues are clarified by the W3C, the VoiceXML Forum will update the test suite.

Observed behavior	ID	Spec. Reference	Abstract
	81	[1.5.2]	If an application root document specifies an application root element error.semantic is thrown.

	588	[6.2.1]	If the http-equiv attribute is set to "expires" and the content attribute is set to "0", the interpreter MUST not cache the document.
	1106	[2.1.6.2.3]	When no <reprompt> is used in conjunction with a <goto nextitem> within an error handler, then the target form item's prompt will not be played.

Untestable assertions

Note: These assertions were not testable due to equipment limitations.

ID	Spec. Reference	Abstract
302	[2.3.7]	If the Interpreter is unable to connect to the destination entity when attempting a transfer because network is busy, the transfer name variable is filled with the value 'network_busy'.
306	[2.3.7.2.2]	If a transfer is terminated by the network between the interpreter and the destination entity, the transfer name variable is filled with the value 'network_disconnect'.

VoiceXML 2.1 Module

Notes

Optional assertions

Tested?	ID	Spec. Reference	Abstract	Reason
Yes	23	[sec-data]	If the name attribute is specified, the interpreter exposes the DOM through the ECMAScript variable corresponding to the value of the name attribute.	
Yes	42	[sec-data-dom]	The DOM exposed by the <data> tag must support the code property of the DOMException object as specified in DOM Level 2 .	
Yes	43	[sec-data-dom]	The DOM exposed by the <data> tag must support the documentElement property of the Document object as specified in DOM Level 2 .	
No	44	[sec-data-dom]	The DOM exposed by the <data> tag must support the getElementsByTagName method of the Document object as specified in DOM Level 2 .	
No	45	[sec-data-dom]	The DOM exposed by the <data> tag must support the getElementsByTagNameNS method of the Document object as specified in DOM Level 2 .	
No	46	[sec-data-dom]	The DOM exposed by the <data> tag must support the getElementById method of the Document object as specified in DOM Level 2 .	
Yes	47	[sec-data-dom]	The DOM exposed by the <data> tag must support the nodeName property of the Node object as specified in DOM Level 2 .	
Yes	48	[sec-data-dom]	The DOM exposed by the <data> tag must support the nodeValue property of the Node object as specified in DOM Level 2 .	
No	49	[sec-data-dom]	The DOM exposed by the <data> tag must support the nodeType property of the Node object as specified in DOM Level 2 .	

Yes	50	[sec-data-dom]	The DOM exposed by the <data> tag must support the parentNode property of the Node object as specified in DOM Level 2 .	
No	51	[sec-data-dom]	The DOM exposed by the <data> tag must support the childNodes property of the Node object as specified in DOM Level 2 .	
Yes	52	[sec-data-dom]	The DOM exposed by the <data> tag must support the firstChild property of the Node object as specified in DOM Level 2 .	
Yes	53	[sec-data-dom]	The DOM exposed by the <data> tag must support the lastChild property of the Node object as specified in DOM Level 2 .	
Yes	54	[sec-data-dom]	The DOM exposed by the <data> tag must support the previousSibling property of the Node object as specified in DOM Level 2 .	
Yes	55	[sec-data-dom]	The DOM exposed by the <data> tag must support the nextSibling property of the Node object as specified in DOM Level 2 .	
No	56	[sec-data-dom]	The DOM exposed by the <data> tag must support the attributes property of the Node object as specified in DOM Level 2 .	
No	57	[sec-data-dom]	The DOM exposed by the <data> tag must support the ownerDocument property of the Node object as specified in DOM Level 2 .	
No	58	[sec-data-dom]	The DOM exposed by the <data> tag must support the namespaceURI property of the Node object as specified in DOM Level 2 .	
No	59	[sec-data-dom]	The DOM exposed by the <data> tag must support the prefix property of the Node object as specified in DOM Level 2 .	
No	60	[sec-data-dom]	The DOM exposed by the <data> tag must support the localName property of the Node object as specified in DOM Level 2 .	
No	61	[sec-data-dom]	The DOM exposed by the <data> tag must support the hasChildNodes method of the Node object as specified in DOM Level 2 .	
No	63	[sec-data-dom]	The DOM exposed by the <data> tag must support the hasAttributes method of the Node object as specified in DOM Level 2 .	
No	64	[sec-data-dom]	The DOM exposed by the <data> tag must support the length property of the NodeList object as specified in DOM Level 2 .	
No	65	[sec-data-dom]	The DOM exposed by the <data> tag must support the item method of the NodeList object as specified in DOM Level 2 .	
No	66	[sec-data-dom]	The DOM exposed by the <data> tag must support the length property of the NamedNodeMap object as specified in DOM Level 2 .	
No	67	[sec-data-dom]	The DOM exposed by the <data> tag must support the getNamedItem method of the NamedNodeMap object as specified in DOM Level 2 .	
No	68	[sec-data-dom]	The DOM exposed by the <data> tag must support the item method of the NamedNodeMap object as specified in DOM Level 2 .	
No	69	[sec-data-dom]	The DOM exposed by the <data> tag must support the getNamedItemNS method of the NamedNodeMap object as specified in DOM Level 2 .	
Yes	70	[sec-data-dom]	The DOM exposed by the <data> tag must support the data property of the CharacterData object as specified in DOM Level 2 .	
No	71	[sec-data-dom]	The DOM exposed by the <data> tag must support the length property of the CharacterData object as specified in DOM Level 2 .	
Yes	72	[sec-data-dom]	The DOM exposed by the <data> tag must support the substringData method of the CharacterData object as specified in DOM Level 2 .	
No	73	[sec-data-dom]	The DOM exposed by the <data> tag must support the name property of the Attr object as specified in DOM Level 2 .	
No	74	[sec-data-dom]	The DOM exposed by the <data> tag must support the specified property of the Attr object as specified in DOM Level 2 .	
No	75	[sec-data-dom]	The DOM exposed by the <data> tag must support the value property of the Attr object as specified in DOM Level 2 .	
No	76	[sec-data-	The DOM exposed by the <data> tag must support the ownerElement	

		dom]	property of the Attr object as specified in DOM Level 2 .	
Yes	77	[sec-data-dom]	The DOM exposed by the <data> tag must support the tagName property of the Element object as specified in DOM Level 2 .	
Yes	78	[sec-data-dom]	The DOM exposed by the <data> tag must support the getAttribute method of the Element object as specified in DOM Level 2 .	
No	79	[sec-data-dom]	The DOM exposed by the <data> tag must support the getAttributeNode method of the Element object as specified in DOM Level 2 .	
No	80	[sec-data-dom]	The DOM exposed by the <data> tag must support the getElementsByTagName method of the Element object as specified in DOM Level 2 .	
No	81	[sec-data-dom]	The DOM exposed by the <data> tag must support the getAttributeNS method of the Element object as specified in DOM Level 2 .	
No	82	[sec-data-dom]	The DOM exposed by the <data> tag must support the getAttributeNodeNS method of the Element object as specified in DOM Level 2 .	
No	83	[sec-data-dom]	The DOM exposed by the <data> tag must support the getElementsByTagNameNS method of the Element object as specified in DOM Level 2 .	
Yes	84	[sec-data-dom]	The DOM exposed by the <data> tag must support the hasAttribute method of the Element object as specified in DOM Level 2 .	
No	85	[sec-data-dom]	The DOM exposed by the <data> tag must support the hasAttributeNS method of the Element object as specified in DOM Level 2 .	
No	86	[sec-data-dom]	The DOM exposed by the <data> tag must support the target property of the ProcessingInstruction object as specified in DOM Level 2 .	
No	87	[sec-data-dom]	The DOM exposed by the <data> tag must support the data property of the ProcessingInstruction object as specified in DOM Level 2 .	
Yes	102	[sec-reco_reco]	<record> may record user utterances while attempting recognition	
No	112	[sec-transfer]	If type is specified and bridge not specified, type takes precedence over the default value of the bridge attribute.	
No	113	[sec-transfer]	If type="consultation" an outgoing call is attempted to the specified destination.	
No	114	[sec-transfer]	If type="consultation" and the caller disconnects while the outgoing call is attempted, connection.disconnect.hangup is thrown.	
No	115	[sec-transfer]	If type="consultation" and the caller cancels the transfer attempt while the outgoing call is attempted by using a DTMF command, the transfer form item variable is filled with near_end_disconnect.	
No	116	[sec-transfer]	If type="consultation" and the caller cancels the transfer attempt while the outgoing call is attempted by using a voice command, the transfer form item variable is filled with near_end_disconnect.	
No	117	[sec-transfer]	If type="consultation" and an outgoing call is attempted to a specified destination that is busy, the transfer form item variable will be filled with busy.	
No	118	[sec-transfer]	If type="consultation" and an outgoing call is attempted to a specified destination but an intermediate network is busy, the transfer form item variable will be filled with network_busy.	
No	119	[sec-transfer]	If type="consultation" and the outgoing call is not answered within the time specified by connecttimeout, the transfer attempt terminates, and the transfer form item variable is filled with noanswer.	
No	120	[sec-transfer]	If type="consultation" and the transfer attempt ends but the reason is not known, the transfer form item variable is filled with unknown.	
No	121	[sec-	If type="consultation" and the outgoing call is answered, the platform	

		transfer]	disconnects from the conversation, connection.disconnect.transfer is thrown, the transfer form item variable is undefined, and the calling and called parties remain connected.	
No	122	[sec-transfer]	If type="consultation" and the platform does not support consultation transfer, error.unsupported.transfer.consultation is thrown.	
No	123	[sec-transfer]	The transfer element's "connecttimeout" value is not honored unless either "bridge" is set to true or "type" is set to "bridge" or "consultation".	
No	124	[sec-transfer]	Exactly one of "bridge" or "type" may be specified; otherwise an error.badfetch event is thrown.	
No	125	[sec-transfer]	The "maxtime" value will only be honored if the "bridge" attribute is set to "true", or the "type" is set to "bridge"	
No	126	[sec-transfer]	When "transferaudio" is specified, and the audio file has not completed playing, the audio will cease upon a far-end connection.	
No	130	[sec-reco_reco]	If recordutterance property is set to false, recording shadow variable will be set to undefined after terminating a <record> with speech	
Yes	131	[sec-reco_reco]	If recordutterance property is set to false, recordingsize shadow variable will be set to undefined after termininating a <record> using speech	
Yes	132	[sec-reco_reco]	If recordutterance property is set to false, recordingduration shadow variable will be set to undefined after terminating a <record> using speech	
No	133	[sec-reco_reco]	If recordutterance property is set to false, recording shadow variable will be set to undefined after terminating a <transfer> using speech	
No	134	[sec-reco_reco]	If recordutterance property is set to false, recordingsize shadow variable will be set to undefined after terminating a <transfer> using speech	
No	135	[sec-reco_reco]	If recordutterance property is set to false, recordingduration shadow variable will be set to undefined after terminating a <transfer> using speech	
No	146	[sec-reco_reco]	<transfer> may record user utterances while attempting recognition	
Yes	157	[sec-data-dom]	The DOM exposed by the <data> tag must support the data property of the Comment object as specified in DOM Level 2 .	
No	158	[sec-data-dom]	The DOM exposed by the <data> tag must support the length property of the Comment object as specified in DOM Level 2 .	
Yes	159	[sec-data-dom]	The DOM exposed by the <data> tag must support the substringData method of the Comment object as specified in DOM Level 2 .	
Yes	160	[sec-data-dom]	The DOM exposed by the <data> tag must support the data property of the Text object as specified in DOM Level 2 .	
No	161	[sec-data-dom]	The DOM exposed by the <data> tag must support the length property of the Text object as specified in DOM Level 2 .	
Yes	162	[sec-data-dom]	The DOM exposed by the <data> tag must support the substringData method of the Text object as specified in DOM Level 2 .	
Yes	163	[sec-data-dom]	The DOM exposed by the <data> tag must support the data property of the CDATASection object as specified in DOM Level 2 .	
No	164	[sec-data-dom]	The DOM exposed by the <data> tag must support the length property of the CDATASection object as specified in DOM Level 2 .	
Yes	165	[sec-data-dom]	The DOM exposed by the <data> tag must support the substringData method of the CDATASection object as specified in DOM Level 2 .	
No	166	[sec-data-dom]	The DOM exposed by the <data> tag must support the nodeName property of the Entity Relationship Node object as specified in DOM Level 2 .	
No	167	[sec-data-dom]	The DOM exposed by the <data> tag must support the nodeValue property of the Entity Relationship Node object as specified in DOM Level	

			2 .	
No	168	[sec-data-dom]	The DOM exposed by the <data> tag must support the nodeType property of the Entity Relationship Node object as specified in DOM Level 2 .	
No	169	[sec-data-dom]	The DOM exposed by the <data> tag must support the parentNode property of the Entity Relationship Node object as specified in DOM Level 2 .	
No	170	[sec-data-dom]	The DOM exposed by the <data> tag must support the childNodes property of the Entity Relationship Node object as specified in DOM Level 2 .	
No	171	[sec-data-dom]	The DOM exposed by the <data> tag must support the firstChild property of the Entity Relationship Node object as specified in DOM Level 2 .	
No	172	[sec-data-dom]	The DOM exposed by the <data> tag must support the lastChild property of the Entity Relationship Node object as specified in DOM Level 2 .	
No	173	[sec-data-dom]	The DOM exposed by the <data> tag must support the previousSibling property of the Entity Relationship Node object as specified in DOM Level 2 .	
No	174	[sec-data-dom]	The DOM exposed by the <data> tag must support the nextSibling property of the Entity Relationship Node object as specified in DOM Level 2 .	
No	175	[sec-data-dom]	The DOM exposed by the <data> tag must support the attributes property of the Entity Relationship Node object as specified in DOM Level 2 .	
No	176	[sec-data-dom]	The DOM exposed by the <data> tag must support the ownerDocument property of the Entity Relationship Node object as specified in DOM Level 2 .	
No	177	[sec-data-dom]	The DOM exposed by the <data> tag must support the hasChildNodes method of the Entity Relationship Node object as specified in DOM Level 2 .	
No	178	[sec-data-dom]	The DOM exposed by the <data> tag must support the hasAttributes method of the Entity Relationship Node object as specified in DOM Level 2 .	
No	200	[sec-transfer]	(SIP version of assertion 113) If type="consultation" an outgoing call is attempted to the specified destination.	
No	201	[sec-transfer]	(SIP version of assertion 119) If type="consultation" and the outgoing call is not answered within the time specified by connecttimeout, the transfer attempt terminates, and the transfer form item variable is filled with noanswer.	

Required assertions

ID	Spec. Reference	Abstract
1	[sec-grammar_expr]	Grammar element may include a srcexpr attribute to dynamically generate grammar URL
2	[sec-grammar_expr]	If both src and srcexpr are included, error.badfetch is thrown
3	[sec-grammar_expr]	If both srcexpr and inline grammar are included, error.badfetch is thrown
4	[sec-grammar_expr]	If none of src, srcexpr nor inline grammar are included, error.badfetch is thrown

5	[sec-grammar_expr]	srcexpr must be evaluated every time the grammar element needs to be executed
7	[sec-grammar_expr]	If srcexpr cannot be evaluated, an error.semantic is thrown
8	[sec-grammar_expr]	If both src and inline grammar are included, error.badfetch is thrown
9	[sec-script_expr]	Script element may include an srcexpr attribute to dynamically generate script URL
10	[sec-script_expr]	If both src and srcexpr are included, error.badfetch is thrown
11	[sec-script_expr]	If none of src, srcexpr nor inline script are included, error.badfetch is thrown
12	[sec-script_expr]	srcexpr must be evaluated every time the script element needs to be executed
13	[sec-script_expr]	If both srcexpr and inline script are included, error.badfetch is thrown
14	[sec-script_expr]	If both src and inline script are included, error.badfetch is thrown
15	[sec-script_expr]	If srcexpr cannot be evaluated, an error.semantic is thrown
16	[sec-mark]	When a <mark> is executed during the processing of a form item, the interpreter sets shadow variables, the names of which correspond to the properties of the application.lastresult$ object. The value of each shadow variable must be identical to the value of the corresponding application.lastresult$ property.
17	[sec-mark]	VoiceXML 2.1 extends the <mark> element to support the nameexpr attribute, which is an ECMAScript expression that evaluates to the name of the <mark>.
18	[sec-mark]	Exactly one of "name" and "nameexpr" must be specified; otherwise, an error.badfetch event is thrown.
19	[sec-mark]	The markname and marktime properties of the application.lastresult$ object must be set whenever the application.lastresult$ object is assigned and a <mark> has been executed.
20	[sec-mark]	If no <mark> was executed, the markname and marktime properties of application.lastresult$ are undefined.
21	[sec-data]	If content is not returned within the specified fetchtimeout, an error.badfetch event is thrown.
22	[sec-data]	The interpreter fetches the URI specified by the src attribute.
24	[sec-data]	If the name attribute is present, and the interpreter doesn't support DOM, the interpreter must throw error.unsupported.data.name.
25	[sec-data]	The interpreter evaluates the srcexpr attribute as an ECMAScript expression when the <data> element needs to be executed. The result of evaluating the srcexpr attribute is the URI to be fetched.
26	[sec-data]	If srcexpr cannot be evaluated, error.semantic is thrown.
27	[sec-data]	Exactly one of "src" or "srcexpr" must be specified; otherwise, an error.badfetch event is thrown.
28	[sec-data]	Interpreters must support the methods GET and POST when submitting an HTTP request. GET is the default method.
29	[sec-data]	Specifying a URL that points to a non-existent resources causes the interpreter to throw a catchable error.badfetch event.
30	[sec-data]	If the name attribute is specified, the interpreter supports the DOM, and the interpreter retrieves an XML document that is not well-formed, the interpreter must throw a catchable error.badfetch.
31	[sec-data]	The <data> element can occur in executable content or as a child of <form> or <vxml>.
32	[sec-data]	The <data> element shares the same scoping rules as the <var> element.
33	[sec-data]	If a <data> element has the same name as a variable already declared in the same scope, the variable is assigned a reference to the DOM exposed by the <data> element.
34	[sec-data]	If use of the DOM causes a DOMException to be thrown, but the DOMException is not caught by an ECMAScript catch handler, the VoiceXML interpreter throws error.semantic.

35	[sec-data]	When an ECMAScript variable is submitted to the server its value is first converted into a string before being submitted.
36	[sec-data]	The default encoding type of the submitted document is "application/x-www-form-urlencoded".
37	[sec-data]	If the enctype attribute is set to "multipart/form-data", the interpreters must submit the document using that encoding type.
38	[sec-data]	If the namelist is not specified, no variables are submitted.
39	[sec-data]	If <data> has a namelist attribute, all and only those variables are submitted.
40	[sec-data]	If a namelist is supplied, it may contain individual variable references which are submitted with the same qualification used in the namelist.
41	[sec-data]	If specified, fetchaudio plays during a long fetch.
88	[sec-foreach]	The <foreach> element can occur as a child of a <prompt> element.
89	[sec-foreach]	The array attribute must be an ECMAScript expression that evaluates to an array; otherwise an error.semantic is thrown.
90	[sec-foreach]	The item variable stores each array item upon iteration of the loop. A new variable will be declared if it is not already defined within the parent's scope.
92	[sec-reco_reco]	When the "recordutterance" property is set to true, the "recording" shadow variable of an input item (e.g. field) stores a reference to the recording when audio is collected from the user. Utterance recordings can be played back using the expr attribute of the audio element.
94	[sec-reco_reco]	when property "recordutterance" is set to true, an input item has been filled and has its shadow variables assigned, "recordingsize" must be set in the shadow variable for that input item variable
95	[sec-reco_reco]	When the shadow variable "recordingsize" is set, it must contain the size of the recorded audio in bytes
96	[sec-reco_reco]	when property "recordutterance" is set to true, an input item has been filled and has its shadow variables assigned, "recordingduration" must be set in the shadow variable for that input item variable
97	[sec-reco_reco]	When the shadow variable "recordingduration" is set, it must contain the duration of the recorded audio in milliseconds
98	[sec-reco_reco]	application.lastresult$ must have the shadow variable "recording" set to the same value as the form item shadow variable
99	[sec-reco_reco]	application.lastresult$ must have the shadow variable "recordingsize" set to the same value as the form item shadow variable
100	[sec-reco_reco]	application.lastresult$ must have the shadow variable "recordingduration" set to the same value as the form item shadow variable
101	[sec-reco_reco]	In the case of <link> and <menu>, the interpreter only sets the application.lastresult$ properties.
103	[sec-reco_reco]	enctype must be "multipart/form-data" when sending the recording on the namelist
104	[sec-reco_reco]	type must be "POST" when sending the recording on the namelist
105	[sec-reco_reco-type]	set the media format of recorded audio during recognition using the recordutterancetype property
106	[sec-reco_reco-type]	The platform must support the audio file format 'audio/basic' as specified in Appendix E of the VoiceXML 2.0 specification.
108	[sec-disconnect]	The default for namelist of a disconnect element is to return no variables; this means the interpreter context will receive an empty ECMAScript object.
109	[sec-disconnect]	The variables specified in the namelist attribute of the disconnect element are returned to the interpreter context. (The precise mechanism by which these variables are made available to the interpreter context is platform specific.)
111	[sec-disconnect]	If the namelist attribute of a disconnect element contains one or more undeclared variables, the interpreter throws error.semantic.

127	[sec-reco_reco]	If recordutterance property is set to false, recording shadow variable will be set to undefined after filling a <field> using speech input
128	[sec-reco_reco]	If recordutterance property is set to false, recordingsize shadow variable will be set to undefined after filling a <field> using speech input
129	[sec-reco_reco]	If recordutterance property is set to false, recordingduration shadow variable will be set to undefined after filling a <field> using speech input
136	[sec-reco_reco]	Recorded utterance may be posted using <submit>
137	[sec-reco_reco]	Recorded utterance may be posted using <subdialog>
138	[sec-reco_reco]	Recorded utterance may be posted using <data>
141	[sec-mark]	If nameexpr cannot be evaluated, an error.semantic event is thrown.
142	[sec-data]	If the datafetchhint property is set to "safe", content of that type is never fetched until it is needed
143	[sec-data]	A cached data resource must be reloaded if the datamaxage property for its type is less than its current age.
144	[sec-data]	A cached data resource must be reloaded if the datamaxstale property for its type is less than its current staleness.
145	[sec-disconnect]	The <disconnect> namelist and the <exit> namelist are processed independently. If the interpreter executes both a <disconnect> namelist and an <exit> namelist, both sets of variables are available to the interpreter context. (The precise mechanism by which these variables are made available to the interpreter context is platform specific.)
147	[sec-reco_reco-type]	platform must support audio file format audio/x-alaw-basic from VXML 2.0
148	[sec-reco_reco-type]	platform must support audio file format audio/x-wav from VXML 2.0
149	[sec-foreach]	The array attribute must evaluate to an ECMAScript Array; otherwise, an error.semantic event is thrown. (Tests various data types from foreach appearing within a block.)
150	[sec-foreach]	The array attribute must evaluate to an ECMAScript Array; otherwise, an error.semantic event is thrown. (Tests various data types from foreach appearing within a prompt in a block.)
151	[sec-foreach]	The array attribute must evaluate to an ECMAScript Array; otherwise, an error.semantic event is thrown. (Tests various data types from foreach appearing within a prompt in a field.)
152	[sec-foreach]	The assigned value could be undefined for a sparse array. (Tests sparse array handling.)
153	[sec-foreach]	The foreach element operates on a shallow copy of the array specified by the array attribute.
154	[sec-foreach]	A shallow copy of the corresponding array element is assigned to the item variable.
155	[sec-foreach]	Executable content is not allowed within foreach contained within a prompt. (This tests a prompt inside block.)
156	[sec-foreach]	Executable content is not allowed within foreach contained within a prompt (this test includes the prompt in a field).

Assertions in Dispute

Note: These assertions identify inconsistencies in VoiceXML or SRGS. As such, there is no "correct" behavior. The observed behavior is indicated here for informational purposes only for the benefit of the application developer. If and when these issues are clarified by the W3C, the VoiceXML Forum will update the test suite.

Observed behavior	ID	Spec. Reference	Abstract

Untestable assertions

Note: These assertions were not testable due to equipment limitations.

ID	Spec. Reference	Abstract

SRGS 1.0 Module

Notes

Optional assertions

Tested?	ID	Spec. Reference	Abstract	Reason
Yes	55	[AppE]	DTMF grammar with alternatives and sequences (external)	
No	56	[AppE]	DTMF grammar with alternatives and sequences (external)	
Yes	57	[AppE]	DTMF grammar with alternatives and sequences (inline)	
No	58	[AppE]	DTMF grammar with alternatives and sequences (inline)	
Yes	59	[AppE]	DTMF grammar with "pound" and "star" as symbols (external)	
No	60	[AppE]	DTMF grammar with "pound" and "star" as symbols (external)	
Yes	61	[AppE]	DTMF grammar with "pound" and "star" as symbols (inline)	
No	62	[AppE]	DTMF grammar with "pound" and "star" as symbols (inline)	
Yes	63	[AppE]	simple DTMF grammar with a sequence (external)	
No	64	[AppE]	simple DTMF grammar with a sequence (external)	
Yes	65	[AppE]	simple DTMF grammar with a sequence (inline)	
No	66	[AppE]	simple DTMF grammar with a sequence (inline)	
Yes	67	[AppE]	simple DTMF grammar with one character (external)	
No	68	[AppE]	simple DTMF grammar with one character (external)	
Yes	69	[AppE]	simple DTMF grammar with one character (inline)	
No	70	[AppE]	simple DTMF grammar with one character (inline)	
Yes	95	[4.5]	language declaration ignored for mode="dtmf" (external)	
No	96	[4.5]	language declaration ignored for mode="dtmf" (external)	
Yes	97	[4.5]	language declaration ignored for mode="dtmf" (inline)	
No	98	[4.5]	language declaration ignored for mode="dtmf" (inline)	
No	103	[4.5]	other language declaration for mode="voice" (external)	
No	104	[4.5]	other language declaration for mode="voice" (external)	
No	105	[4.5]	other language declaration for mode="voice" (inline)	
No	106	[4.5]	other language declaration for mode="voice" (inline)	
Yes	131	[4.6]	mode declaration dtmf (external)	
No	132	[4.6]	mode declaration dtmf (external)	

Yes	133	[4.6]	mode declaration dtmf (inline)	
No	134	[4.6]	mode declaration dtmf (inline)	

Required assertions

ID	Spec. Reference	Abstract
1	[2.4]	alternative reference to NULL (external)
2	[2.4]	alternative reference to NULL (external)
3	[2.4]	alternative reference to NULL (inline)
4	[2.4]	alternative reference to NULL (inline)
5	[2.4]	one-of element containing a single item (external)
6	[2.4]	one-of element containing a single item (external)
7	[2.4]	one-of element containing a single item (inline)
8	[2.4]	one-of element containing a single item (inline)
9	[2.4]	alternative reference to a single tag (external)
10	[2.4]	alternative reference to a single tag (external)
11	[2.4]	alternative reference to a single tag (inline)
12	[2.4]	alternative reference to a single tag (inline)
13	[2.4]	a set of alternatives with a legal weight on every alternative (external)
14	[2.4]	a set of alternatives with a legal weight on every alternative (external)
15	[2.4]	a set of alternatives with a legal weight on every alternative (inline)
16	[2.4]	a set of alternatives with a legal weight on every alternative (inline)
17	[2.4]	a set of alternatives with no weights (external)
18	[2.4]	a set of alternatives with no weights (external)
19	[2.4]	a set of alternatives with no weights (inline)
20	[2.4]	a set of alternatives with no weights (inline)
21	[2.4]	a single alternative without a weight (external)
22	[2.4]	a single alternative without a weight (external)
23	[2.4]	a single alternative without a weight (inline)
24	[2.4]	a single alternative without a weight (inline)
25	[2.4]	a single alternative with a weight (external)
26	[2.4]	a single alternative with a weight (external)
27	[2.4]	a single alternative with a weight (inline)
28	[2.4]	a single alternative with a weight (inline)
29	[2.4]	a set of alternatives with legal weights on one or many but not all alternatives (external)
30	[2.4]	a set of alternatives with legal weights on one or many but not all alternatives (external)
31	[2.4]	a set of alternatives with legal weights on one or many but not all alternatives (inline)
32	[2.4]	a set of alternatives with legal weights on one or many but not all alternatives (inline)
33	[4.9]	base uri declaration (external)
34	[4.9]	base uri declaration (external)
35	[4.9]	base uri declaration (inline)
36	[4.9]	base uri declaration (inline)

37	[4.9.1]	base uri precedence (external)
38	[4.9.1]	base uri precedence (external)
39	[4.9.1]	base uri precedence (inline)
40	[4.9.1]	base uri precedence (inline)
41	[4.12]	comments (external)
42	[4.12]	comments (external)
43	[4.12]	comments (inline)
44	[4.12]	comments (inline)
45	[5.4]	activate root, single public rule, set of public rules or roots of one or many grammars directly or indirectly referenced (same language) (external)
46	[5.4]	activate root, single public rule, set of public rules or roots of one or many grammars directly or indirectly referenced (same language) (external)
47	[2.1]	basic tokens (external)
48	[2.1]	basic tokens (external)
49	[5.4]	inform hosting environment if unable to process content (external)
50	[5.4]	inform hosting environment if unable to process content (inline)
71	[2.2]	duplicate rulenames (external)
72	[2.2]	duplicate rulenames (inline)
73	[2.2]	redefining special rulenames (external)
74	[2.2]	redefining special rulenames (inline)
75	[AppJ.1]	Example 1 XML (external)
76	[AppJ.1]	Example 1 XML (external)
77	[AppJ.1]	Example 1 XML (inline)
78	[AppJ.1]	Example 1 XML (inline)
79	[AppJ.2]	Example 2 booking (external)
80	[AppJ.2]	Example 2 booking (external)
81	[]	example 2 booking (external)
82	[]	example 2 booking (external)
83	[AppJ.2]	Example 2 places (external)
84	[AppJ.2]	Example 2 places (external)
85	[AppJ.2]	Example 2 places (inline)
86	[AppJ.2]	Example 2 places (inline)
87	[3.3]	example phrases (external)
88	[3.3]	example phrases (external)
89	[3.3]	example phrases (inline)
90	[3.3]	example phrases (inline)
91	[4.4]	no character encoding (external)
92	[4.4]	no character encoding (external)
93	[4.4]	no character encoding (inline)
94	[4.4]	no character encoding (inline)
99	[4.5]	language declaration of en-US for mode="voice" (external)
100	[4.5]	language declaration of en-US for mode="voice" (external)
101	[4.5]	language declaration of en-US for mode="voice" (inline)
102	[4.5]	language declaration of en-US for mode="voice" (inline)
123	[4.11]	meta declarations (external)

124	[4.11]	meta declarations (external)
125	[4.11]	HTTP response headers (external)
126	[4.11]	HTTP response headers (external)
127	[4.11]	HTTP response headers (inline)
128	[4.11]	HTTP response headers (inline)
129	[4.11]	meta declarations (inline)
130	[4.11]	meta declarations (inline)
135	[4.6]	no mode declaration implies (external)
136	[4.6]	no mode declaration implies (external)
137	[4.6]	no mode declaration implies (inline)
138	[4.6]	no mode declaration implies (inline)
139	[4.6]	mode declaration (external)
140	[4.6]	mode declaration (external)
141	[4.6]	mode declaration (inline)
142	[4.6]	mode declaration (inline)
143	[4]	XML grammar with no DOCTYPE (external)
144	[4]	XML grammar with no DOCTYPE (external)
145	[4]	XML grammar with no DOCTYPE (inline)
146	[4]	XML grammar with no DOCTYPE (inline)
149	[4]	Grammar with no version (external)
150	[4]	Grammar with no version (inline)
151	[2.5]	zero repetitions never matched (external)
152	[2.5]	zero repetitions never matched (external)
153	[2.5]	zero repetitions never matched (inline)
154	[2.5]	zero repetitions never matched (inline)
155	[2.5]	any number of NULL's is equivalent to a single NULL (external)
156	[2.5]	any number of NULL's is equivalent to a single NULL (external)
157	[2.5]	any number of NULL's is equivalent to a single NULL (inline)
158	[2.5]	any number of NULL's is equivalent to a single NULL (inline)
159	[2.5]	repeat an expansion between "n" and "m" times (external)
160	[2.5]	repeat an expansion between "n" and "m" times (external)
161	[2.5]	repeat an expansion between "n" and "m" times (inline)
162	[2.5]	repeat an expansion between "n" and "m" times (inline)
163	[2.5]	repeat an expansion "m" or more times (external)
164	[2.5]	repeat an expansion "m" or more times (external)
165	[2.5]	repeat an expansion "m" or more times (inline)
166	[2.5]	repeat an expansion "m" or more times (inline)
167	[2.5]	2.5 repeat an expansion exactly "n" times (external)
168	[2.5]	2.5 repeat an expansion exactly "n" times (external)
169	[2.5]	2.5 repeat an expansion exactly "n" times (inline)
170	[2.5]	2.5 repeat an expansion exactly "n" times (inline)
171	[2.5]	optional expansion (external)
172	[2.5]	optional expansion (external)
173	[2.5]	optional expansion (inline)

174	[2.5]	optional expansion (inline)
175	[2.5]	optional VOID equivalent to NULL (external)
176	[2.5]	optional VOID equivalent to NULL (external)
177	[2.5]	optional VOID equivalent to NULL (inline)
178	[2.5]	optional VOID equivalent to NULL (inline)
179	[2.5]	repeat probabilities (external)
180	[2.5]	repeat probabilities (external)
181	[2.5]	repeat probabilities (inline)
182	[2.5]	repeat probabilities (inline)
183	[4.7]	root rule declaration (external)
184	[4.7]	root rule declaration (external)
185	[4.7]	root rule declaration (inline)
186	[4.7]	root rule declaration (inline)
187	[3.1]	rule definition (external)
188	[3.1]	rule definition (external)
189	[3.1]	rule definition (inline)
190	[3.1]	rule definition (inline)
191	[3.1]	empty item expansion (external)
192	[3.1]	empty item expansion (external)
193	[3.1]	empty item expansion (inline)
194	[3.1]	empty item expansion (inline)
195	[3.1]	no empty expansions (external)
196	[3.1]	no empty expansions (inline)
197	[3.1]	NULL expansion (external)
198	[3.1]	NULL expansion (external)
199	[3.1]	NULL expansion (inline)
200	[3.1]	NULL expansion (inline)
201	[3.2]	declaration of private rule (external)
202	[3.2]	declaration of private rule (external)
203	[3.2]	declaration of private rule (inline)
204	[3.2]	declaration of private rule (inline)
205	[3.2]	declaration of public rule (external)
206	[3.2]	declaration of public rule (external)
207	[3.2]	declaration of public rule (inline)
208	[3.2]	declaration of public rule (inline)
209	[3.2]	external reference to private root rule (external)
210	[3.2]	external reference to private root rule (external)
211	[3.2]	declaration of private rule (external)
212	[3.2]	declaration of private rule (external)
213	[2.2.2]	reference to the root rule of a grammar identified by a URI with a Media Type (external)
214	[2.2.2]	reference to the root rule of a grammar identified by a URI with a Media Type (external)
215	[2.2.2]	reference to the root rule of a grammar identified by URI with a Media Type (inline)
216	[2.2.2]	reference to the root rule of a grammar identified by URI with a Media Type (inline)
217	[2.2.2]	reference to a named rule of a grammar identified by a URI (external)

218	[2.2.2]	reference to a named rule of a grammar identified by a URI (external)
219	[2.2.2]	reference to a named rule of a grammar identified by URI (inline)
220	[2.2.2]	reference to a named rule of a grammar identified by URI (inline)
221	[2.2.2]	reference to a named rule of a grammar identified by a URI with a Media Type (external)
222	[2.2.2]	reference to a named rule of a grammar identified by a URI with a Media Type (external)
223	[2.2.2]	reference to a named rule of a grammar identified by a URI with a Media Type (inline)
224	[2.2.2]	reference to a named rule of a grammar identified by a URI with a Media Type (inline)
225	[2.2.1]	local rule reference (external)
226	[2.2.1]	local rule reference (external)
227	[2.2.1]	local rule reference (inline)
228	[2.2.1]	local rule reference (inline)
229	[2.2.2]	media type specified does not match actual content (external)
230	[2.2.2]	media type specified does not match actual content (inline)
231	[2.2.2]	modes do not agree between rulerefs (external)
232	[2.2.2]	reference to grammar with mismatched mode
233	[2.2]	reference to a non-existent local rule reference (external)
234	[2.2]	reference to a non-existent local rule reference (inline)
235	[3.1]	single tag expansion (external)
236	[3.1]	single tag expansion (external)
237	[3.1]	single tag expansion (inline)
238	[3.1]	single tag expansion (inline)
239	[2.3]	empty item element (external)
240	[2.3]	empty item element (external)
241	[2.3]	empty item element (inline)
242	[2.3]	empty item element (inline)
243	[2.3]	element containing only whitespace (external)
244	[2.3]	element containing only whitespace (external)
245	[2.3]	element containing only whitespace (inline)
246	[2.3]	element containing only whitespace (inline)
247	[2.3]	sequence of rule references (external)
248	[2.3]	sequence of rule references (external)
249	[2.3]	sequence of rule references (inline)
250	[2.3]	sequence of rule references (inline)
251	[2.3]	sequence of tokens and rule references (external)
252	[2.3]	sequence of tokens and rule references (external)
253	[2.3]	sequence of tokens and rule references (inline)
254	[2.3]	sequence of tokens and rule references (inline)
255	[2.3]	sequence of tokens (external)
256	[2.3]	sequence of tokens (external)
257	[2.3]	sequence of tokens (inline)
258	[2.3]	sequence of tokens (inline)
259	[2.2.3]	special rule definitions ($GARBAGE) (external)
260	[2.2.3]	special rule definitions ($GARBAGE) (external)
261	[2.2.3]	special rule definitions ($GARBAGE) (inline)

262	[2.2.3]	special rule definitions ($GARBAGE) (inline)
263	[2.2.3]	special rule definitions ($NULL) (external)
264	[2.2.3]	special rule definitions ($NULL) (external)
265	[2.2.3]	special rule definitions ($NULL) (inline)
266	[2.2.3]	special rule definitions ($NULL) (inline)
267	[2.2.3]	special rule definitions ($VOID) (external)
268	[2.2.3]	special rule definitions ($VOID) (external)
269	[2.2.3]	special rule definitions ($VOID) (inline)
270	[2.2.3]	special rule definitions ($VOID) (inline)
271	[4.8]	tag format declaration (external)
272	[4.8]	tag format declaration (external)
273	[4.8]	tag format declaration (inline)
274	[4.8]	tag format declaration (inline)
275	[2.6]	any number of tags may be included (external)
276	[2.6]	any number of tags may be included (external)
277	[2.6]	any number of tags may be included (inline)
278	[2.6]	any number of tags may be included (inline)
283	[2.6]	standalone tags are legal (external)
284	[2.6]	standalone tags are legal (external)
285	[2.6]	standalone tags are legal (inline)
286	[2.6]	standalone tags are legal (inline)
287	[2.1]	basic tokens (external)
288	[2.1]	basic tokens (external)
289	[2.1]	basic tokens (inline)
290	[2.1]	basic tokens (inline)
291	[2.1]	quoted tokens (external)
292	[2.1]	quoted tokens (external)
293	[2.1]	quoted tokens (inline)
294	[2.1]	quoted tokens (inline)
295	[2.1]	Unicode tokens (external)
296	[2.1]	Unicode tokens (external)
297	[2.1]	Unicode tokens (inline)
298	[2.1]	Unicode tokens (inline)
299	[4.7]	reference to undefined root rule (external)
300	[4.7]	reference to undefined root rule (inline)
305	[4.7]	grammar ref with no fragment and no root rule (external)
306	[]	Cannot refer to an undefined rule without fragment with a uri; referring grammar (external)
307	[]	duplicated special rule names (inline)
308	[]	duplicated special rule names (inline)
309	[]	Need to add description and spec reference
310	[]	Need to add description and spec reference
311	[]	Need to add description and spec reference
312	[]	Need to add description and spec reference
315	[]	Need to add description and spec reference

316		Need to add description and spec reference

Assertions in Dispute

Note: These assertions identify inconsistencies in VoiceXML or SRGS. As such, there is no "correct" behavior. The observed behavior is indicated here for informational purposes only for the benefit of the application developer. If and when these issues are clarified by the W3C, the VoiceXML Forum will update the test suite.

Observed behavior	ID	Spec. Reference	Abstract

Untestable assertions

Note: These assertions were not testable due to equipment limitations.

ID	Spec. Reference	Abstract