

ICS 33.030

M 21

YD

中华人民共和国通信行业标准

YD/T 2098-2010

信息无障碍 语音上网技术要求

Information accessibility technical requirement of
voice-enabled web service

2010-12-29 发布

2011-01-01 实施

中华人民共和国工业和信息化部 发布

目 次

前 言.....II

1 范围.....1

2 规范性引用文件.....1

3 术语、定义及缩略语.....1

4 概述.....4

5 语音上网服务系统结构.....5

6 设备功能要求.....6

7 Voice XML 系统架构.....8

8 语音标记语言格式.....8

9 语音浏览器与语音服务器信息交互格式.....35

10 语音服务系统安全性要求.....40

附录 A（资料性附录） Voice XML 应用示例.....42

前 言

本标准为“信息无障碍”系列标准之一，该系列标准预计的结构及名称如下。

——基础类：

信息无障碍 术语、符号和命令

——对现有服务系统的补充：

YD/T 1761-2008 信息无障碍 身体机能差异人群 网站设计无障碍技术要求

YD/T 1822-2008 信息无障碍 身体机能差异人群 网站设计无障碍评级测试方法

YD/T 2065-2009 信息无障碍 用于身体机能差异人群的通信终端设备设计导则

YD/T 1890-2009 信息终端设备信息无障碍辅助技术的要求和评测方法

信息无障碍 呼叫中心服务系统技术要求

——专用服务系统：

信息无障碍 公众场所内听力障碍人群辅助系统技术要求

——专用设备：

YD/T 1889-2009 手柄电话助听器耦合技术要求和测量方法

移动电话助听器耦合要求和测量方法

骨导电话机传输性能的研究

——专用技术

用于手语和唇读的低比特视频通信应用

信息无障碍 语音上网技术要求

本标准规定的网页设计的主要技术原则与万维网联盟（W3C）制定 Voice XML 2.0 及 Voice XML2.1 保持了一致。

本标准的附录 A 是资料性附录。

本标准由中国通信标准化协会提出并归口。

本标准起草单位：工业和信息化部电信研究院、中国残疾人联合会信息中心、中国互联网协会、全国老龄工作委员会办公室、中国盲文出版社、华为技术有限公司。

本标准主要起草人：吴英桦、崔慧萍、刘 盈、孙永革、吴玉韶、沈 静、杨 崑、虞庆平、何 川、黄 畅。

信息无障碍 语音上网技术要求

1 范围

本标准规定了利用语音方式访问互联网的技术要求，包括语音上网服务系统结构、设备功能要求、Voice XML 系统架构、语音标记语言格式、语音浏览器与语音服务器信息交互格式、语音服务系统安全性要求等。

本标准适用于语音上网服务系统及相关设备。

2 规范性引用文件

下列文件中的条款通过本标准的引用而成为本标准的条款。凡是注日期的引用文件，其随后所有的修改单（不包括勘误的内容）或修订版均不适用于本标准。然而，鼓励根据本标准达成协议的各方研究是否可使用这些文件的最新版本。凡是不注日期的引用文件，其最新版本适用于本标准。

IETF RFC 2046	多用途互联网邮件扩展第二部分：媒体类型
IETF RFC 2392	内容标识和消息标识统一资源定位符
IETF RFC 2396	统一资源定位符：通用句法
IETF RFC 2616	超文本传送协议
IETF RFC 2822	互联网消息格式
IETF RFC 3550	实时传送协议
IETF RFC 4463	媒体资源控制协议
W3C CSS	层叠样式表
W3C SRGS	语音识别语法规则

3 术语、定义及缩略语

3.1 术语和定义

下列术语和定义适用于本标准。

3.1.1

语音扩展标记语言 Voice Extensible Markup Language (VoiceXML、VXML)

VoiceXML 是一种标记语言，用来创建音频对话，主要包括语音合成、数字化音频、语音识别、DTMF 按键输入识别、录音等交互式会话。它的主要作用是把基于网络的开发和信息这两者的优势引入语音应答系统。

VoiceXML 有以下优点。

- 通过在每个文档中指定多个交互式对话，最大限度地减少客户机和服务器之间的交互。
- 使得程序员不用理会底层的和平台特有的细节。
- 使得用户交互的代码（在 VoiceXML 中）和业务逻辑（例如 CGI 脚本）分离。
- 提高业务在不同平台的可移植性。VoiceXML 对内容提供商、工具提供商和平台提供商来说是一

种通用的语言。

- 它可以很容易地应用到简单的交互中，也可通过提供一些语言特性来支持复杂的对话。

VoiceXML 描述的是由语音应答系统提供的人机交互，包括以下几个方面。

- 语音合成 (Text-to-Speech);
- 声音文件的输出;
- 语音输入的认识;
- DTMF 输入的识别;
- 对话流的控制;
- 电话的一些特性，如呼叫转移和挂机。

鉴于语音上网的特殊要求，对电话特性的支持不在本标准涉及范围内。

3.1.2

VXML 标记语言中的术语和定义

3.1.2.1

文档 Document

一个 VoiceXML 文档构成了一个描述会话的有限状态机。用户每次只能处于一个会话状态或 Dialog 中。每个 Dialog 都会确定要跳转的下一个 Dialog。跳转通过 URI 指定，URI 规定了下一个要用到的文档和 Dialog。如果 URI 没有指向一个文档，则认为它指向当前文档。如果 URI 没有指向一个 Dialog，则认为它指向那个文档的第一个 Dialog。如果一个 Dialog 没有指定它的下一个 Dialog，或者它有一个明确地退出会话的元素，则执行中断。

3.1.2.2

对话 Dialogs

VoiceXML 有两种 Dialog，即 Form 和 Menu。Form 定义了一个收集用户输入，并给相应的 Field 变量赋值的交互过程。每个 Field 可以指定一个语法，这个语法规定了 Field 允许的输入。如果有 Form 级的语法存在，一个输入语句可能同时填充几个 Field 变量。Menu 给用户提供了一些可选的选项，并根据用户的选择跳转到另外一个 Dialog。

3.1.2.3

子对话 SubDialog

Subdialog 就像函数调用一样，它提供了一种机制来调用一个新的交互，并返回到调用它的 Form。调用前的变量实例、语法和状态信息都被保存起来，在返回到发起调用的文档后仍然可用。例如，Subdialog 可以用在创建一个由数据库查询得到的确定序列；或者创建在单个应用的文档中共享的组件；或者在多个应用中可重用的 Dialog 库。

3.1.2.4

会话 Session

会话从用户开始和 VoiceXML 解释器环境交互时开始，囊括了加载和执行文档的过程，并随着用户或文档或解释器环境请求结束而结束。

3.1.2.5

应用 Application

应用是由共享同一应用根文档的一系列文档组成的。无论如何，只要用户在跟应用里的文档交互，它的根文档都会被加载。当用户在同一应用的其他文档中跳转时，应用根文档一直都被加载，直到用户跳转到一个不在这个应用里的文档。当应用根文档被加载的时候，它的变量就像这个应用的变量一样，对这个应用里所有的文档都是可用的，它的语法在这个应用的生命期内都是激活的。

3.1.2.6

语法 Grammar

由于语音识别技术的现状，非特定人领域自由无限制的语音输入无法得到令人满意的识别精度，因此人机语音交互系统中，一般采用有限制的语法规则对语音输入进行一定限制。

每个 Dialog 都有一个或多个语音和（或）DTMF 语法。在机器主导的应用中，每个 Dialog 的语法只有当用户在那个 Dialog 中的时候才激活。而在混合主动式（Mixed Initiative）的应用中，相关的几个 Dialog 的语法同时都是激活的（也就是正在监听），即使用户在同一个文档的另外一个 Dialog，或者在同一个应用的另一个加载的文档。混合主动式是指，用户和机器交替地决定下一步要做的。在这种情况下，如果用户说的话匹配了另一个 Dialog 中激活的语法，执行平台会跳转到那个 Dialog，用户刚才所说的话就像它在那个 Dialog 中说的一样。混合主动式增强了语音应用程序的灵活性和性能。

3.1.2.7

事件 Event

平台能在多种环境下抛出事件，例如当用户没有响应的时候，或者没有清楚地响应的时候，或者要求帮助的时候等。如果发现 VoiceXML 文档有语义错误的时候，也会抛出事件。事件由 Catch 元素捕捉。

3.1.2.8

链接 Link

链接支持混合主动式交互。它可以指定一个语法，无论何时，只要用户在这个 Link 的作用域内，这个语法都是激活的。如果用户的输入匹配了这个 Link 的语法，控制就跳转到该 Link 指定的 URI。

3.1.3

媒体资源控制协议 Media Resource Control Protocol (MRCP)

MRCP 用于管理和访问分布式系统架构上的语音资源服务器，采用 MRCP 协议后，独立软件商和应用开发商仅需面向 MRCP 接口撰写程序，而无需顾及语音识别和其电话应用方面的差异，不必为不同语音提供商的语音引擎分别撰写程序，任何支持 MRCP 标准的语音引擎都可以被无缝集成和调用。

3.1.4

MRCP 中的术语和定义

3.1.4.1

请求 Request

指代 MRCP 客户端发送给 MRCP 服务器端的控制请求。

3.1.4.2

响应 Response

指代 MRCP 服务器端返回给 MRCP 客户端的响应

3.1.4.3

事件 Event

指代 MRCP 服务器端返回给 MRCP 客户端的异步事件，用于通知关键事件的发生。

3.1.4.4

方法 Method

指代 MRCP 方法，包含语音合成方法，语音识别方法等。不同的方法，对应于不同的语音合成 / 识别功能的调用。

3.1.4.5

消息头 Message Header

指代 MRCP 消息（包括请求、响应、事件）的头部，通常包含一个或者多个参数。

3.1.4.6

消息体 Message Body

指代 MRCP 消息（包括请求、响应、事件）的内容。

3.1.4.7

返回码 Status Code

指代 MRCP 响应包含的返回码，用于确定 MRCP 请求是否成功完成。

3.2 缩略语

下列缩略语适用于本标准。

ASR	Automatic Speech Recognition	自动语音识别
DTMF	Dual Tone Multi-Frequency	双音多频
IVR	Interactive Voice Response	交互式语音应答
ICP	Internet Content Provider	互联网内容提供商
MRCP	Media Resource Control Protocol	媒体资源控制协议
PDA	Personal Digital Assistant	个人数字助理
PLMN	Public Land Mobile Network	公共陆地移动网
PSTN	Public Switched Telephone Network	公共电话交换网
RTP	Transport Protocol for Real-Time Applications	实时传送协议
SRGS	Speech Recognition Grammar Specification	语音识别语法规范
TTS	Text to Speech	语音合成
VOIP	Voice Over Internet Protocol	互联网协议电话
XML	Extensible Markup Language	可扩展标记语言

4 概述

语音上网技术是指通过语音控制的方式访问互联网，利用语音操控网页的浏览，收听网络信息。

语音上网技术可解决残疾人、老年人面临的上网困难。传统的上网方式离不开键盘、鼠标和屏幕，输出的信息显示在屏幕上，盲人、弱视者、肢残者等特殊人群使用这种上网方式比较困难。在使用手机上网的情况下，狭小的显示屏也会影响老年人阅读。语音上网可以解决这些问题，利用语音识别技术将信息输入终端，利用语音合成或其他音频输出方式播放信息，避免了视觉障碍带来的信息交流困难。

语音上网技术可解决低收入人口上网缺乏计算机终端的问题。目前人们从互联网获取各种资源时，主要是借助 PC 机，而实际上，PC 机的普及率远远低于电话。语音上网技术的实现，使更多的人能够有机会

访问互联网，以电话拨号方式接入，通过电话按键或语音输入来访问互联网上的各种资源并存取数据。

语音上网技术不仅可以方便特殊人群上网，还可应用到车载系统、手机、PDA 等产品中，使得普通人在不方便用眼的时候也可以上网，例如在驾驶车辆的同时，通过车载系统访问网站，或利用手机、各种随身的微型通信设备实现无键盘上网。

语音浏览器和语音服务器是构成语音上网服务系统的基本要素，语音浏览器可以实现连接电话网与互联网的语音网关上，也可以实现在手机、车载设备、PDA 等各类用户终端上；语音服务器由服务提供商支持。用户访问语音上网服务系统时，可以通过两种方式，一是通过电话网访问，二是通过支持语音上网功能的手机、车载设备等访问网站，如图 1 所示。通过电话网访问语音网站时，需要在电话网和 IP 网之间设置语音网关，语音网关支持语音浏览器、语音识别、语音合成功能，可以根据用户的声音或者 DTMF 按键命令识别用户请求，浏览器将其转化为语音标记语言，发往服务器并获取服务器提供的音频文件或文字信息，音频文件被直接播放，文字信息被转化为语音播放。通过手机、车载设备访问网站时，终端除了要支持浏览器功能外，还要支持语音识别功能，通过声音命令控制网页的浏览，并把从服务器下载的音频文件播放给用户听，或者利用语音合成技术把下载的文字转化成声音播放。

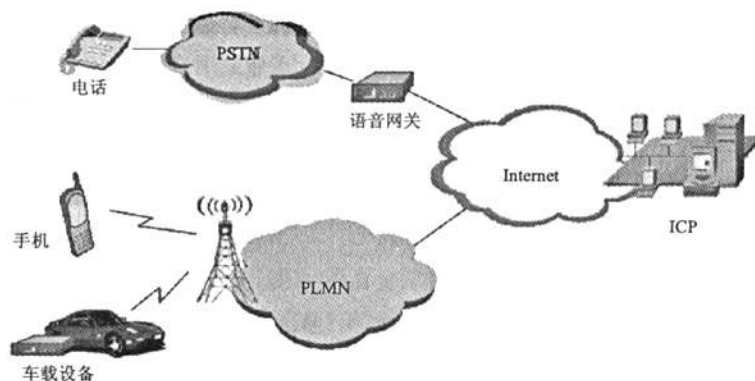


图 1 语音上网服务系统构成示意

5 语音上网服务系统结构

语音上网的系统包括语音门户、ASR、TTS、IVR 服务、语音网关、运营管理等模块，如图 2 所示。

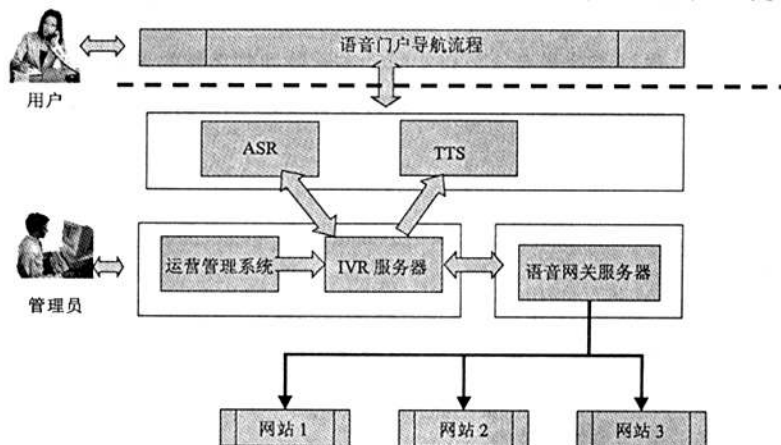


图 2 语音上网服务系统结构

1) 语音门户

语音门户提供了语音上网的门户和通道,用户语音上网时,可以听到语音门户生成的语音导航菜单。用户只需记住该某个号码就可以通过导航菜单进入后续的上网。语音门户还可以用来完成个性化收藏夹管理,即用户将几个语音门户中的栏目设置为收藏夹,下次使用时,系统将根据其收藏夹生成菜单,这样用户就可以快速进入自己喜欢的网站了。

2) TTS

语音合成指的是计算机自动地把给定的文本信息转换成语音的过程,它赋予了计算机系统“嘴巴”功能——按照文本发音。可以将网站的内容播报给用户收听。语音合成是复杂的语音处理技术,是涉及语音学、语言学、数字信号处理和计算机科学等领域的多学科综合性技术。语音合成技术把可视的文本信息转换为可闻的声音信息,其应用范围非常广,如文本的有声校对、电脑话务员系统、信息自动广播系统、语音应答系统、信息库查询系统、文本校对系统以及残疾人辅助发音系统等。

3) ASR

语音识别是一个通过模式识别匹配方法将人的语音转换为文本的过程。用户可以通过语音方式来进行上网的操作,比如选择某个网站等。ASR技术提供了计算机“耳朵”功能,计算机可以通过语音的感知接收命令。在这个过程中,计算机首先要根据人的语音特点建立语音模型,对输入的语音信号进行分析,并抽取所需的特征,在此基础上建立语音识别所需的模板。而计算机在识别过程中要根据语音识别的整体模型,将计算机中存放的语音模板与输入的语音信号的特征进行比较,根据一定的搜索和匹配策略,找出一系列最优的与输入的语音匹配的模板,然后,据此模板号的定义,通过查表就可以给出计算机的识别结果。

4) IVR 服务器

提供对自动流程的控制,在自动流程的运行过程中,将会调用收号资源、语音资源和TTS/ASR资源。完成上网服务的基本业务流程。

5) 语音网关服务

与互联网的接口服务器,IVR服务器与语音网关配合,完成用户上网浏览过程。语音网关采用VXML脚本方式,用于存放各个网站的服务地址,并可以向某网站发送浏览请求。

6) 运维管理系统

配置子系统、监控子系统、报表子系统等,通过管理系统可以以配置平台的资源、进行日常维护、动态加载或卸载网站业务。通过监控子系统可以实时动态地监视到系统的各种设备的运行数据和状态。通过报表子系统可以得到系统的话务量、服务质量以及业务代表工作情况的统计报表。通过告警子系统可以查看系统的告警信息。

6 设备功能要求

6.1 语音网关功能要求

通过电话网提供语音上网服务时,语音网关作为前端接入设备,通过数字中继/光纤/IP接入承载通道,采用7号信令方式与传统PSTN网络连接,或采用SIP信令方式与NGN网络连接,为用户提供语音服务。IVR主要负责完成业务流程解释、管理、业务呼叫的处理,提供语音播放、TTS合成、ASR、DTMF的接收和发送,语音网关的访问和业务话单的产生等。

语音网关把PSTN线路承载的语音信息,转化为数据,并在互联网上传播。语音网关与互联网上的语

音网站对接，向互联网上的语音网站发送HTTP请求，获取语音网站的内容，并支持网站的切换等功能。

互联网的语音网站为特殊的网站，需要用语音标记语言编写语音网页文件。语音网关的一次服务的过程有以下4个步骤。

1) 语音门户等待电话用户的接入，如果有电话连接到语音门户中，IVR平台就会通知语音网关，并将相关信息通知语音网关；

2) 语音网关向语音网站发起请求，网站的文档服务器获取语音标记语言文档；

3) 语音网关核心解释文档语义，根据语义和用户发生交互，其中包括通过平台向用户播放语音文件或者播放文本合成语音，获得用户的按键输入或者语音输入；

4) 文档结束，解释器核心结束工作。

6.2 语音服务器功能要求

语音服务器是实现语音上网的关键要素，其功能要求分为3个等级：

1) 等级1

提供基本的语音识别、语音合成功能。

语音识别用于把上网用户的语音指令识别成文本内容，而语音上网站点可以根据识别出的文本，进行业务逻辑的控制。

语音合成用于把语音上网站点提供的文本内容合成为声音文件并输出，使得上网用户可以通过电话/手机/PDA/车载系统等设备进行网页浏览。

2) 等级2

支持媒体资源控制协议（见 IETF RFC 4463），支持媒体资源控制方式，可发起语音识别/语音合成请求，控制语音识别/语音合成操作，获取语音识别/语音合成结果。

语音服务器可同时支持多个并发的语音识别/语音合成请求，并通过实时传输协议（RTP，见 IETF RFC 3550）进行输入/输出语音的传送，适用于语音上网环境。

3) 等级3

提供说话人身份认证功能。

语音服务器可额外提供说话人认证功能满足语音上网站点的安全性需求。

说话人身份认证功能根据说话人声音特征的差异性，对上网用户进行身份确认，从而提供信息安全保障。

6.3 语音浏览器功能要求

语音浏览器也是实现语音上网的基本要素之一，其功能要求分为3个等级。

1) 等级1

支持 VoIP，接收上网用户的输入语音，并输出语音合成的声音。

语音上网有多种接入手段，但最终语音数据需要在网络上进行高速、高质量传播，以保证实时的语音识别/语音合成服务。

语音浏览器是整个语音上网系统的对外接口，负责声音的输入与输出。会话初始协议（SIP）可以帮助语音浏览器对并发的语音输入/输出请求进行有效的控制和处理，从而提供对 VoIP 的支持。

2) 等级2

支持解析并执行语音标记语言的功能。语音标记语言用于定义语音上网站点的业务逻辑，语音浏览

器负责对语音标记语言进行解析并执行。

3) 等级

支持媒体资源控制协议 (MRCP)，语音浏览器与语音服务器的交互通过 MRCP 完成。语音浏览器解析语音标记语言时，如果遇到请求语音输入或者请求语音输出的语句时，浏览器负责把该请求转化为 MRCP 指令，发送给语音服务器，并接受语音服务器递交的结果。

7 Voice XML 系统架构

Voice XML 服务系统由文档服务器、Voice XML 解释器环境和实现平台三个功能模块构成，其架构如图 3 所示。

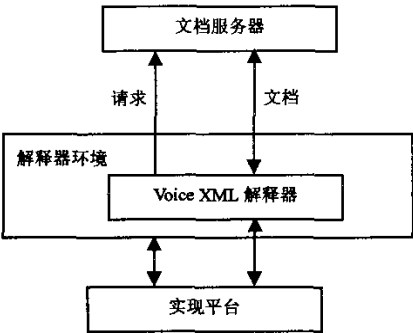


图 3 Voice XML 系统架构

文档服务器（例如 Web 服务器）通过 VoiceXML 解释器环境（Context）处理来自客户端应用程序，即 VoiceXML 解释器（语音浏览器）的请求，并把相应的 VoiceXML 文档返回给 VoiceXML 解释器处理。

通过 VoiceXML 解释器环境，VoiceXML 解释器会同时监听多个用户的行为。例如，在一个用户教程中，VoiceXML 解释器可能会一直监听一段特别的提示输出，以便更好的引导用户，而同时另一个被监听用户则正在进行参数的改变，如音量或者语音合成的特性。

执行平台由 VoiceXML 解释器环境和 VoiceXML 解释器控制。例如，在交互式语音应答系统中，VoiceXML 解释器环境负责检测电话呼入，若有则把电话接起来并把这次呼叫引导到程序的第一个文档；而 VoiceXML 解释器在电话接起来后负责控制对话框。执行平台则产生一些事件来响应用户相应的操作（如接收到的语音或字符输入，或者挂机）和相应的系统事件（如定时器到期）。

8 语音标记语言格式

8.1 Voice XML 元素

支持语音上网的网页应符合 Voice XML 标记语言的格式，VoiceXML 是用来创建音频对话的，支持人机交互的语音合成、数字化音频、语音识别、DTMF 按键输入识别、录音等互动式会话功能。Voice XML 是基于 XML 的标记语言，遵循 XML 的基本语法规则，在语音上网应用中，需要使用的 VoiceXML 元素见表 1。

表 1 VoiceXML 元素

元 素	作 用
<Assign>	给变量赋值
<Audio>	在 Prompt 中播放一段音频
<Block>	没有人机交互操作的可执行代码的容器
<Catch>	事件捕获
<Choice>	定义一个 Menu Item
<Clear>	清除一个或多个 Form Item 变量
<Else>	用于<i>元素中的 Else
<Elseif>	用于<i>元素中的 Elseif
<Enumerate>	列举 Menu 中的 Choice 的信息
<Error>	捕获<error>事件
<Exit>	退出会话
<Field>	在 Form 中声明一个输入域
<Filled>	在输入域被填充后执行一些操作
<Form>	用于给出信息和收集数据的 Dialog
<Goto>	在同一或不同文档中跳转
<Grammar>	指定语音识别或 DTMF 语法
<Help>	捕获<help>事件
<i>	简单的条件逻辑
<Initial>	在进入一个混合主动式的 Form 时声明初始的逻辑
<Link>	对所有在 Link 的作用域内的会话指定一个跳转
<Log>	生成调试信息
<Menu>	提供可供选择的跳转
<Meta>	以 Name/Value 对的形式定义一个元数据项
<Metadata>	使用元数据方案定义元数据信息
<Noinput>	捕获<Noinput>事件
<Nomatch>	捕获<Nomatch>事件
<Object>	与自定义的扩展功能进行交互
<Option>	在<Field>中指定一个可选项
<Param>	<Object>或<SubDialog>的参数。
<Prompt>	产生一个输出给用户的语音合成和音频的队列
<Property>	控制平台的设置
<Record>	录音
<Reprompt>	在某个事件被捕获后重新访问 Field 时，再次播放这个 Field 的 Prompt
<Return>	从 Subdialog 返回
<Script>	指定一段客户端的 ECMAScript 脚本逻辑
<Subdialog>	在当前的 Dialog 中调用另外一个 Dialog 作为 Subdialog
<Submit>	提交一些值到文档服务器
<Throw>	抛出事件
<Value>	在 Prompt 中插入一个表达式的值
<Var>	声明一个变量
<Vxml>	在每个 VoiceXML 文档中最上层的元素

8.2 对话结构元素

VoiceXML 中与对话逻辑相关联的元素，包括<Form>、<Field>、<Filled>、<Menu>等。

8.2.1 Form

Form 是 VoiceXML 文档的主要组成部分，它包括以下几个部分：

● 一组 Form Item，即在 FIA (Form Interpretation Algorithm) 的主循环被访问的一些元素。Form Item 可细分为 Input item 和 Control Item，Input Item 可由用户填充，而 Control Item 不能；

- 非 Form Item (Non-Form Item) 变量的声明；
- 事件处理；
- 填充后 (Filled) 的操作，也就是当某些 Input Item 的组合被赋值后要执行的程序逻辑模块。

Form 元素的属性见表 2。

表 2 Form 元素的属性

ID	Form 的名字。如果有指定的话，这个 Form 就可以在该文档内被引用，或者在别的文档中引用它。例如，<form id="weather">，<goto next="#weather">
Scope	该 Form 语法的默认作用域 (Scope)。如果它的值为 Dialog，则该 Form 语法只是在这个 Form 里是激活的；如果它的值为 Document，则该 Form 语法在这个文档的任何对话中都是激活的；如果该文档为应用根文档，则该 Form 语法在这个应用的任何文档的任何 Dialog 中都是激活的。注意，单独的 Form 的语法作用域的优先权高于默认的语法作用域。例如在非根文档的某个 Form 默认的作用域为 Dialog，且有一个 Form 语法的作用域为 Document，则该语法在这个文档的任何对话中都是激活的

Form 中可以嵌入其他元素，可以分为输入元素 (Input Item) 和控制元素 (Control Item) 两类。输入元素指定可以得到用户输入的变量。它有一些提示，告诉用户要说或者键入什么，定义允许用户输入的语法，处理一些由此产生的事件的事件处理。

输入元素的类型见表 3。

表 3 输入元素类型

<Field>	它的值是通过 ASR 或 DTMF 语法获得的
<Record>	它的值是用户录音的一段音频。例如，<Record>元素可以用来收集语音邮件的语音信息
<Object>	它通过一些参数调用依赖于平台的 Object，该 Object 是一个 ECMAScript Object。平台的 Object 可以是一个内置的 Dialog，比如说用来收集信用卡信息的 Dialog。也可以是用特定的 DTMF 文本方法收集文本消息的 Dialog。虽然当依赖于平台的 Object 不被平台支持时，要抛出 error.unsupported.objectname 事件来处理<Object>元素，但是对于这些依赖于平台的 Object 的执行却是没有任何限制的
<Subdialog>	<Subdialog>类似于函数调用。它调用同一文档中的另一个 Dialog，或者调用另一个文档。它返回的是一个 ECMAScript Object

输入元素一般又可以包含下面几个元素：

- <Filled>元素，它包含了一些在相应的 Input Item 被填充后要执行的操作。
- <Property>元素，用于指定作用于该 Input Item (<Initial>元素也可以包含<Property>元素) 的 Property。
- <Prompt>元素，用于指定相应的元素被访问时要播放的提示语。
- <Grammar>元素，用于指定该 Input Item (<Subdialog>不能包含<Grammar>元素) 允许输入的语音或 DTMF 按键。
- <Catch>元素及它的一些简写形式，用于指定作用于该 Input Item (<Initial>元素不能包含<Catch>

元素)的事件处理。

控制元素中不需要收集用户输入，有 2 种，见表 4。

表 4 控制元素类型

<Block>	它包含一些程序语句的序列，这些程序语句是用于提示和计算的，不是用于收集输入的。<Block>有一个 Form Item 变量（通常是隐含的），在它被解释之前，该变量被置为 True
<Initial>	该元素用于在混合主动式的 Form 里控制最初的交互。它应该提示用户说一些话来匹配一个 Form 级别的语法。在<Initial>元素的执行中，当至少有一个 Input Item 变量被识别结果所填充时，<Initial>元素的 Form Item 变量变为 True，而后就不能再次被解释器执行

8.2.2 <Field>

<Field>是一个 Form，用于收集用户输入，它的属性见表 5。

表 5 Field 元素的属性

Name	该 Form Item 的 Form Item 变量，它的作用域为 Dialog，它保存了识别的结果，在所在的 Form 的 Form Item 中，该 Form Item 变量名应是唯一的，如果该变量名不是唯一的，在获取文档时会抛出 error.badfetch 事件，该变量名应符合变量的命名规则
Expr	该 Form Item 变量的初始值，默认为 ECMAScript 的 undefined。如果给它一个初始值，该 Form Item 将不会被访问，除非它被清零
Cond	一个表达式，当它的结果值为 true 时，该 Form Item 才被访问。如果没有指定该属性值，也能被访问
Type	该<Field>的类型，即内置语法类型的名称。平台对内置语法类型的支持是可选的。如果不支持一个指定的内置语法类型，要抛出一个 error.unsupported.builtin 事件
Slot	语法槽名，用于存放变量（如果没有指定，默认值为该变量名）。当所用的语法格式支持返回一组成对的槽/值（Slot/Value）的机制，且槽名与该 Form Item 变量名不同时，该属性就很有用
Modal	如果它的值为 False（默认值），在该 Field 的收集阶段所有激活的语法都可以匹配；如果它的值为 True，则只有该 Field 的语法是允许匹配的，其他的都暂时失效

<Field>元素的影子变量（Shadow Variables）见表 6，该<Field>的变量名为 Name。

表 6 <Field>元素的影子变量

name\$.utterance	被识别的词의原始串。正确的标记和拼写是依赖于平台的（例如“five hundred thirty”或“5 hundred 30”或只是“530”）。如果是 DTMF 语法，该变量将包含匹配的数字串
name\$.inputmode	用户输入的模式，值为 Dtmf 或 Voice
name\$.interpretation	一个 ECMAScript 变量，它包含了用户输入的语义解释
name\$.confidence	该<Field>的 Confidence 级别，它的取值范围为 0.0~1.0，0.0 表示最小的 Confidence，1.0 表示最大的 Confidence。平台可以用 Utterance 的 Confidence（application.lastresult\$.confidence 的值）作为 name\$.confidence 的值，<Field>和 Utterance 级别的 Confidence 之间的差别是依赖于平台的。Confidence 值更详细的解释是依赖于平台的，因为它的计算方法很可能每个平台不一样

通过影子变量，开发者可以对业务逻辑进行更加细致的控制。

<Filed> 中可以采用<Grammar>或者<Option>对用户可输入的可选项进行控制。

8.2.3 <Record>

<Record>元素是一个 Form 的输入元素，用于收集用户的录音。

<Record>元素的属性见表 7。

表 7 Record 元素的属性

Name	该 Input Item 变量保存了所录制的音频。注意，各个平台实现该变量的方式可以是不同的（虽然该技术说明书中要求所有的平台都应在<Audio>元素和<Submit>元素中支持该变量的行为）值
Expr	该 Form Item 变量的初始值，它默认为 ECMAScript 的 Undefined 。如果给它赋一个初始值，则该 Form Item 就不会被访问，除非该 Form Item 变量被清零
Cond	ECMAScript 表达式。只有当其结果值为 boolean true，该元素才被执行，否则不被执行。缺省时为 True
Modal	如果该属性值为 True（默认值），在录音时，所有非本地的语音和 DTMF 语法都会失效。若为 False，则是激活的
Beep	当为 True 时，录音前有“嘟”音提示。缺省值为 False
Maxtime	最长录音时间。它是一个 Time Designation。默认为一个依赖于平台的值
Finalsilence	表示录音结束的静音输入时间。它是一个 Time Designation。默认为一个依赖于平台的值
Dtmfterm	如果该属性值为 True，任何没有匹配一个激活语法的 DTMF 按键被认为匹配了一个激活的（Anonymous）本地 DTMF 语法。默认值为 True
Type	表示所录制的音频的媒体格式

VoiceXML 要求业务平台支持的播放和录音的音频格式见表 8。

表 8 VoiceXML 要求支持的音频格式

音频格式	媒体类型
Raw (headerless) 8kHz 8-bit mono mu-law [PCM] single channel. (G.711)	audio/basic
Raw (headerless) 8kHz 8 bit mono A-law [PCM] single channel. (G.711)	audio/x-alaw-basic
WAV (RIFF header) 8kHz 8-bit mono mu-law [PCM] single channel.	audio/x-wav
WAV (RIFF header) 8kHz 8-bit mono A-law [PCM] single channel.	audio/x-wav

“Audio/Basic” Mime Type 通常使用 “au” 头格式和无头的 8-bit 8kHz mu-law 的格式。如果该 Mime Type 用于录音，则应使用 mu-law 格式。为了能够播放 “Audio/Basic” Mime Type，平台应支持 mu-law 格式，也可以支持 “au” 格式。

在录音之后，<Record>元素的影子变量也被赋值见表 9。

表 9 <Record>元素的影子变量赋值

name\$.duration	录音的持续时间，单位为毫秒
name\$.size	录音的大小，单位为 Byte
name\$.termchar	如果该 Form Item 的 Dtmfterm 属性值为 True，且用户通过 DTMF 按键终止录音，则该影子变量的值为用户的按键（例如“#”），否则它的值为 Undefined
name\$.maxtime	如果是因为达到最大录音时间而终止录音，则该变量值为 True，否则为 False

8.2.4 <Object>

<Object>是 Form 的输入元素。在 VoiceXML 应用程序中通过<Object>元素，可以扩展出更多的功能（例如可以对 Java 程序包进行调用）。

在初始化和执行期间，<Object>元素直接使用它自己的内容（例如它的<Param>子元素）。和其他的 Field Item 一样，<Object>元素也可以使用<Prompt>元素和<Catch>元素，它也可以有<Filled>操作。

<Object>元素的属性见表 10。

表 10 <Object>元素的属性

Name	当该<Object>元素执行时，该 Form Item 变量被置为一个 ECMAScript Object 类型的值
Expr	该 Form Item 变量的初始值，默认为 ECMAScript 的 Undefined。如果它的值不为 ECMAScript 的 Undefined，则它将不会被访问，除非该 Form Item 变量被清零
Cond	ECMAScript 表达式。只有当其结果值为 Boolean True，该 Form Item 块才被执行，否则不被访问。缺省时为 True
Classid	一个 URI，它指定了要执行的 Object 的位置。该 URI 的拼写规则是依赖于平台的
Codebase	基路径，用于解析 Classid、Data 和 Archive 属性指定的相对 URI。它默认为当前文档的基础 URI
Codetype	下载 classid 指定的 Object 时期望的数据类型。如果没有指定，默认为属性 type 的值
Data	指定 Object 数据的位置的 URI。如果它是一个相对的 URI，则以 Codebase 属性指定的值为基路径
Type	Data 属性指定的数据的类型
Archive	一组和该 Object 有关的档案的 URI，由空格隔开。这些档案包含了 Classid 和 Data 属性指定的数据。如果这些 URI 是一个相对的 URI，则以 Codebase 属性指定的值为基路径
Fetchhint	它默认为<Property>的 Documentfetchhint 的值
Fetchtimeout	它默认为<Property>的 Fetchtimeout 的值
Maxage	它默认为<Property>的 Documentmaxage 的值
Maxstale	它默认为<Property>的 Documentmaxstale 的值

8.2.5 <Subdialog>

Subdialog 是 Form 一类输入元素。Subdialog 提供了重用相同的 Dialog 和建立可重用应用库的一种机制。

发起调用的 Dialog 中的<subDialog>元素通过它的属 Src 或 Srcexpr 调用了被调用的 Dialog（即 Subdialog）。Subdialog 在一个新的执行环境中执行，该执行环境包括了该 Subdialog 中所有的声明和状态信息，它的文档，应用根文档（如果有的话），计数器重置和变量初始化。当执行了一个<Return>元素或<exit>元素，或再没有任何符合条件的 Form Item 可执行时，Subdialog 返回。当 Subdialog 返回时，它的执行环境被删除，且在发起调用的 Dialog 中恢复执行。<Return>元素使控制和数据返回到发起调用的 Dialog 中。

<Subdialog>元素所在的 Dialog 和被调用的 Dialog 的环境是互相独立的，即使这两个 Dialog 是在同一个文档中。在被调用的 Dialog 中是不能访问调用 Dialog 的作用域链中的变量的。在两个执行环境中没有任何共享的变量实例。即使调用 Dialog 和被调用 Dialog 在同一个文档中，它们的执行环境包含的也是不同的变量实例。当调用 Dialog 和被调用 Dialog 在不同的文档中，但它们的根文档是同一个时，Subdialog 中的根变量同样是不同的变量实例。当 Subdialog 返回时，所有应用于 Subdialog 环境的变量绑定都丢失了。

从编程的角度看，Subdialog 的工作机制和子程序的工作机制是不一样的，因为它的调用和被调用的环境是相互独立的，Subdialog 不能访问调用它的 Dialog 中的变量实例，而子程序取可以访问调用程序中的变量实例。类似的，Subdialog 在语言上不遵循事件渗透模式（Event Percolation Model），而在 Java 语言中，如果在某个方法中抛出事件，且该被调用的环境没有处理该事件，则该事件会自动的渗透到调用的环境中。在 Subdialog 中抛出的事件交由 Subdialog 中的事件处理取处理。这些事件处理要返回到调用的环境中只有通过显式的返回该事件。

<Subdialog>元素的 Src 或 Srcexpr 属性指定了被调用的 Subdialog 的 URI（见 IETF RFC2396）。

<Subdialog>元素的属性见表 11。

表 11 <Subdialog>元素的属性

Name	存放 SubDialog 返回值的变量，它是一个 ECMAScript Object，该对象的属性就是<Return>元素的 Namelist 属性中的变量名
Expr	该 Form Item 变量的初始值，它默认为 ECMAScript 的 Undefined。如果给它赋一个初始值，则该 Form Item 就不会被访问，除非该 Form Item 变量被清零
Cond	ECMAScript 表达式。只有当其结果值为 boolean true，该元素才被执行，否则不被执行。缺省时为 True
Namelist	要提交的变量名列表，默认是提交 0 个变量。如果指定了该属性，它可以包含多个变量，以空格隔开。在 VoiceXML 和 ECMAScript 中声明的变量都可以被提交
Src	要调用的 SubDialog 的 URI
Srcexpr	ECMAScript 表达式，它的结果值为要调用的 Subdialog 的 URI
Method	请求的方法，HTTP GET 或 POST
Enctype	被提交的文档的媒体编码类型，默认为 application/x-www-form-urlencoded。解释器也应支持 multipart/form-data，也可以支持其他的编码类型
Fetchaudio	它默认为<Property>的 Fetchaudio 的值
Fetchhint	它默认为<Property>的 Documentfetchhint 的值
Fetchtimeout	它默认为<Property>的 Fetchtimeout 的值
Maxage	它默认为<Property>的 Documentmaxage 的值
Maxstale	见 6.1 节。它默认为<Property>的 Documentmaxstale 的值

应正确的指定属性 Src 和 Srcexpr 中的一个，否则，要抛出一个 error.badfetch 事件。

<Subdialog>元素不但可以包含所有 Form Item 可包含的元素，还可以使用<Param>元素指定要传给 Subdialog 的参数，这些参数应在 Subdialog 中用<Var>元素声明。

8.2.6 <Filled>

<Filled>元素指定了当一个或多个 Form 输入元素被填充后要执行的操作。它可以出现在两个地方：作为<Form>元素的子元素，或者作为输入元素的子元素。例如：<Filled>元素的属性见表 12。

表 12 <Filled>元素的属性

Mode	值为 All（默认）或 Any。如果值为 any，当任意一个指定的 Input Item 被用户最近一次的输入填充后，就执行该<Filled>元素的操作。如果值为 all，当所有提到的 Input Item 被填充，且用户最后一次输入填充了至少一个 Input Item，就执行该<Filled>元素的操作。在 Input Item 中的<Filled>元素不能指定该属性
Namelist	要触发的 Input Item。对于<Form>元素中的<Filled>元素，该属性默认为该 Form 中所有的 Input Item 变量（包括显式和隐式）。一个 Input Item 中的<Filled>元素则不能指定该属性，这种情况下，Namelist 属性值实际上就是该 Input Item 的名称。注意 Control Item 不能出现在该属性中

8.2.7 <Block>

<Block>是 Form 的一个控制元素，它包含的是可执行的内容。

<Block>的属性见表 13。

表 13 <Block>元素的属性

NAME	该 Form Item 变量名，它用来标记该<Block>是否符合被执行的条件。默认为不可访问的内部变量
Expr	该 Form Item 变量的初始值，默认为 ECMAScript 的 undefined。如果给该 Form Item 变量一个初始值，该 Form Item 就不会被执行，除非它的 Form Item 变量被清零
Cond	它是一个 Boolean 表达式，用来判定该 Form Item 是否可以访问。在会话之后，它的值应为 True

8.2.8 <Initial>

<Initial>是 Form 的一个控制元素。

混合主动式的 Form，即计算机和用户共同主导该会话可以提供更好的用户体验。要完成混合主动式 Form,应有一个或多个Form级别的语法。这种会话的编写有几种方式，其中一种普遍的方式是使用<Initial>元素和<Field>元素，<Initial>元素用来获得全面的信息，而<Field>元素用来获得特定的信息。

<Initial>元素的属性见表 14。

表 14 <Initial>元素的属性

Name	该 Form Item 变量的名称，用于检测该<Initial>元素是否为可选定的。默认为一个不可访问的内部变量
Expr	该 Form Item 变量的初始值。默认为 ECMAScript 的 Undefined。如果赋一个初始值给它，该 Form Item 将不会被访问，除非该 Form Item 变量被清零
Cond	ECMAScript 表达式。只有当其结果值为 boolean true，该代码块才被执行，否则不被执行。缺省时为 True

8.2.9 Menu

Menu 在语法上是一个 Form 的另一种形式，该 Form 包含了单个的匿名的 Field。Menu 提示选择一个选项，并根据该选项跳转到相应的 Dialog。就像一个正规的 Form 一样，Menu 也有它自己的语法作用域，例如当用户在执行另一个 Dialog 时，它的语法也可以是激活的。

<Menu>元素可以标识一个菜单，并确定它的语法的作用域。Menu 的属性见表 15。

表 15 Menu 元素的属性

ID	Menu 的标识符，它允许 Dialog 通过<Goto>或<Submit>跳转到该 Menu
Scope	Menu 的语法作用域。如果它的值为“Dialog”，只有当用户跳转到该 Menu，它的语法才是激活的。如果它的值为“Document”，它的语法在整个文档内（或者，如果该 Menu 在应用根文档里，在该应用中，所有加载的文档内）都是激活的
Dtmf	当它的值为“True”，前 9 个 Choice 如果没有明确的指定 Dtmf 属性，则它们会得到隐含的 dtmf 属性，如“1”、“2”等。剩下的那些没有明确指定 Dtmf 属性的 Choice 不会得到 DTMF 值（因此它们不能通过 DTMF 按键来匹配）。此时，如果该 Menu 有 Choice 指定了它自己的 DTMF 序列，该序列包含除了“*”、“#”或“0”以外的字符，平台要抛出 error.badfetch 事件。该属性默认值为“false”
Accept	当它的值为“exact”（该值为默认值），则该 Menu 中的 Choice 元素的文本定义的是要识别的精确的短语。如果它的值为“Approximate”，则该 Menu 中的 Choice 元素的文本定义的是要识别的近似的短语。<Choice>的 Accept 属性的优先权高于 Menu 的该属性

8.2.10 <Choice>

<Choice>元素仅可使用于<Menu>中，有下面几个作用：

- 它可以指定一个语音语法，用<Grammar>元素定义或由程序自动产生；
- 它可以指定一个 DTMF 语法；
- 它的内容可以作为<Enumerate>的提示语；
- 当某个选项被选择的话，它要么指定了要抛出的事件，要么指定了要跳转的 URI。

<Choice>的属性见表 16。

表 16 <Choice>元素的属性

Dtmf	该<Choice>元素的 DTMF 序列。它的作用等同于一个简单的 DTMF 语法，或者应用于该序列识别的 DTMF Property。它和 DTMF 不同的是，它的空格是不起作用的：dtmf="123#"和 dtmf="1 2 3 #"是等同的
Accept	该<Choice>元素的“Accept”属性的优先权高于所在的<Menu>元素的“Accept”属性。当该属性的值为“Exact”（默认），<Choice>元素的文本定义了要识别的精确的短语；当该属性值为“Approximate”，<Choice>元素的文本定义的是要识别的近似的短语
Next	要跳转的下一个对话或文档
Expr	一个表达式，它的值为要跳转的 URI，而不指定“Next”属性
Event	指定一个要抛出的事件，而不指定“Next”属性
Eventexpr	一个 ECMAScript 表达式，它的值为要抛出的事件名
Message	一个信息串，它提供了关于要抛出的事件的上上下文。在<Catch>元素的作用域内用变量“_message”可以得到该信息串
Messageexpr	一个 ECMAScript 表达式，它的值为一个信息串
Fetchaudio	默认为<Property>的 Fetchaudio 的值
Fetchhint	默认为<Property>的 Documentfetchhint 的值
Fetchtimeout	默认为<Property>的 Fetchtimeout 的值
Maxage	默认为<Property>的 Documentmaxage 的值
Maxstale	默认为<Property>的 Documentmaxstale 的值

应正确地指定属性“Next”、“Expr”、“Event”和“Eventexpr”中的一个，否则会抛出 error.badfetch 事件。也可以正确地指定属性“Message”或“Messageexpr”中的一个，否则会抛出 error.badfetch 事件。

如果在<Choice>中指定了一个<Grammar>元素，则该外部语法用于替换自动产生的语法。这样开发者就可以准确地控制<Choice>元素的语法。

8.2.11 <Enumerate>

<Enumerate>元素可以自动产生对<choice>元素或者<Option>元素的描述，它指定了一个模板，该模板根据<Choice>元素在<Menu>元素中的顺序，依次应用于每个<Choice>。如果<Enumerate>中没有内容，则使用默认的模板，列出所有的<Choice>。如果<Enumerate>中有内容，这些内容就是模板的样式，它可以引用两个专有变量，“_Prompt”和“_dtmf”，前者表示<Choice>元素的提示语，后者表示<Choice>元素的 DTMF 序列。

8.2.12 <Link>

Link 元素概念上不属于 Dialog 范畴，但可以被包含在<Vxml>和<Form>元素中也可以作为 Form Item<Field>和<Initial>的子元素。

作为<Vxml>的子元素，<Link>元素的语法在整个文档中都是激活的；作为<Form>的子元素，<Link>元素的语法在该 Form 中是激活的；如果应用根文档中有一个文档级的<Link>元素，它的语法在该应用所有被加载的文档中都是激活的。当这些语法中的一个被匹配，<Link>元素被激活，要么跳转到一个新的文档或 Dialog（如<Goto>），要么抛出一个事件（如<Throw>）。

如果在一个 Form 元素中执行，且该元素的 Modal 属性为 True，则 Form 级或文档级的<Link>元素的语法是不激活的。

从概念上讲，可以认为<Link>元素由两部分组成：条件和操作。“条件”就是<Link>元素的内容，即它的语法，只有语法被匹配了，才能激活。“操作”由该元素的属性指定，即要跳转到哪里或抛出哪个事件。

<Link>元素的属性见表 17。

表 17 <Link>元素的属性

Next	要跳转到的 URI，该 URI 是一个文档（也许有段标识符指定一个起始的 Dialog）或当前文档的一个 Dialog
Expr	和 Next 属性一样，只不过该 URI 是根据给定的 ECMAScript 表达式动态地求值的
Event	当用户匹配了<Link>元素中的一个语法时要抛出的事件
Eventexpr	一个 ECMAScript 表达式，它的结果值为当用户匹配了<Link>元素中的一个语法时要抛出的事件
Message	描述事件产生原因的文本信息
Messageexpr	一个 ECMAScript 表达式，它的结果值为描述事件产生原因的文本信息
Dtmf	该<Link>元素的 DTMF 序列，它的作用等同于一个简单的 DTMF 语法，和应用于该序列识别的 DTMF<Property>。它和 DTMF 语法不同的是，它的空格是不起作用的：dtmf="123#"跟 dtmf="1 2 3 #"的效果是一样的。该属性能和其他的<Grammar>同时使用。当用户的输入匹配了<Link>元素中的一个语法或 DTMF 序列，该<Link>元素被激活
Fetchaudio	它默认为 Fetchaudio<Property>的值
Fetchhint	它默认为 Documentfetchhint<Property>的值
Fetchtimeout	它默认为 Fetchtimeout<Property>的值。
Maxage	它默认为 Documentmaxage<Property>的值。
Maxstale	它默认为 Documentmaxstale<Property>的值。

应正确的指定属性“Next”，“Expr”，“Event”或“Eventexpr”中的一个，否则会抛出一个 error.badfetch 事件。

8.3 用户输入

8.3.1 语音语法

对于语音上网等 VoiceXML 应用而言，用户输入（语音或 DTMF 输入）应被正确识别，进而执行正确的业务逻辑。在 VoiceXML 中，用户输入应符合预先定义的语法规则，<Grammar>元素可用于提供一个语音语法。

- 该语法指定了一系列要匹配的短语，用户可能说出这些短语用于执行一个操作或提供一些信息。
- 对于一个要匹配的短语，该语法可以返回一个相应的语义解释。返回的可以是一个简单的值（例如一个字符串），一组成对的属性——值（例如年、月、日），或一个嵌套的对象（对于复杂的请求）。

<Grammar>元素适用于满足上述要求的任意的语法格式。VoiceXML 平台应至少支持一种通用的格式，即 W3C SRGS 的 XML 格式，也应该支持 W3C SRGS 的 ABNF 格式。VoiceXML 平台可以选择支持 SRGS 以外的语法格式，用于对用户输入进行扩展。例如，平台可能借助<Grammar>元素来嵌入一个专有的语法，用于加入一定的自然语言处理的支持，提升语音服务的用户体验水平。

VoiceXML 平台应是一个合格的 XML 格式的语法处理器，符合 W3C SRGS 中的定义。虽然这样使得平台要处理文档中定义的一个或多个“xml:lang”属性，但是并不要求平台应是多语言的。当遇到一种不支持的语言时，平台要抛出 error.unsupported.language 事件，该事件在它的“_message”变量中指定了这种不支持的语言。

<Grammar>元素从 W3C SRGS 继承的属性见表 18。

表 18 <Grammar>元素从 W3C SRGS 继承的属性

Version	它定义了语法的版本号
Xml:lang	该语法的语言标识符（例如，“fr-CA”表示加拿大法语）。如果缺省，将从文档中继承
Mode	其值为 Voice 或 Dtmf。说明该语法是匹配语音的或是匹配 DTMF 的输入
Root	定义该语法的根规则
Tag-Format	定义该语法中所有的<Tag>元素的内容格式
Xml:base	定义一个基本 URI，用于解析该语法中的相对 URI。该属性的优先权高于<Vxml>元素中的相应属性。如果缺省，则从文档中继承

<Grammar>在 VoiceXML 中增加的属性见表 19。

表 19 <Grammar>在 VoiceXML 中增加的属性

Src	指定一个 URI。如果引用一个外部语法，该属性指定了该语法的位置和该语法中的一个规则名（可选的）。该 URI 被作为一个规则引用来解释，详见 SRGS 的 2.2 节。但是，并不是所有的规则引用的形式都被 VoiceXML 允许。规则引用的能力将在后面详细描述
Scope	值为“Document”或“Dialog”。如果为“Document”，则该语法在当前文档（和相关的用户文档）所有的 Dialog 中都是激活的。如果值为“Dialog”，则该语法只在该 Form 中激活。如果缺省，则该语法的作用域由它的父元素决定
Type	该语法的媒体类型。该属性值的优先权高于其他可能存在的媒体类型（例如 HTTP 或 RTSP 交换中的“Content-Type”，或文件扩展名）。如果该属性缺省，解释器环境将试图动态的确定它的媒体类型（例如，使用服务器特定的媒体类型、文件扩展名或内容自检）。如果语法的内容包含在该元素中，且没有指定该属性，就假定该媒体类型为一个 XML 语法。如果该语法源没有包含选定媒体类型的正确内容，当使用该语法时，抛出一个错误 W3C 的 XML 格式语法暂定的媒体类型为“application/srgs+xml”，ABNF 格式语法暂定的媒体类型为“application/srgs”
Weight	指定该语法的 weight
Fetchhint	默认为 Grammarfetchhint Property
Fetchtimeout	默认为 Fetchtimeout Property
Maxage	默认为 Grammarmaxage Property
Maxstale	默认为 Grammarmaxstale Property。

<Grammar>中内嵌子元素定义需遵循 W3C SRGS 规范，在此仅列出其基本元素，见表 20。

表 20 <Grammar>中内嵌子元素

元 素	作 用
<Grammar>	XML 格式语法的根元素
<Meta>	等价于 HTTP 元内容的头部声明
<Metadata>	XML 元数据内容的头部声明
<Lexicon>	发音词典的头部声明
<Rule>	定义一条语法展开规则
<Token>	定义一个可作为输入的词或其他实体
<Ruleref>	引用一条本地或外部定义的规则
<Item>	定义一个可选的、重复的或可能的展开项
<One-of>	定义一组可供选择的规则展开项
<Example>	包含在一个规则定义中的元素，该元素提供了一个匹配该规则的输入的例子
<Tag>	定义一个任意的字符串，该字符串可作为该展开规则的语义解释，由 W3C SISR 规范定义

<Grammar>元素可以定义显示（External）语法、内嵌（Inline）语法、内置（Built-in）语法。

8.3.2 DTMF 语法

<Grammar>元素也可以用来提供 DTMF 语法。

- 它指定了一组按键，用户可以使用这些按键执行一个操作或提供一些信息。
- 对于匹配的 DTMF 输入，它可以返回一个相应的语义解释。返回的可以是一个简单的值（例如一个字符串），一组成对的属性——值（例如年、月、日），或一个嵌套的对象（对于复杂的请求）。

VoiceXML 平台应支持 XML 格式的 DTMF 语法，以提高程序的可移植性。

<Grammar>元素的 Mode 属性用于区分 DTMF 语法和语音语法。<Grammar>元素的 xml:lang 属性在 DTMF 语法的处理中无效的。在其他的方面，对 DTMF 语法和语音语法的处理都是一样的，包括定义内嵌语法，内置语法，或引用外部语法。对于处理媒体类型，作用域和获取也是一样的。

8.3.3 语法作用域

Form 输入元素的语法的作用域为包含该输入元素的元素。包含在输入元素中的<Grammar>不能指定 Scope 属性，否则抛出 error.badfetch 事件。

<Link>元素中的语法的作用域为包含该<Link>元素的元素，因此，如果该<Link>元素是在应用根文档中，则该语法在该应用所有加载的文档中都是激活的。包含在<Link>元素中的<Grammar>不能指定 Scope 属性，否则抛出 error.badfetch 事件。

<Form>元素的语法的作用域为 Dialog。因此只有用户在该 Form 中时，语法才是激活的。如果此时<Grammar>元素的 Scope 属性值为 Document，则当用户在该文档中时，该语法也是激活的。如果<Grammar>元素的 Scope 属性值为 Document，且该文档为应用根文档，则无论用户在该应用加载的哪一个文档中，这些语法都是激活的。有两种方式可以使 Form 中的语法的作用域为 Document：一种是指定<Form>元素的 Scope 属性为 Document，另一种是指定<Grammar>元素的 Scope 属性为 Document。如果两个元素都指定了 Scope 属性，则以<Grammar>元素中指定的为准。

<Menu>元素中的语法默认作用域为 Dialog，只有用户在该<Menu>元素中，它的语法才被激活。但是也可以指定这些语法的作用域为 Document，使它们在整个文档中都是激活的。如果该文档是应用根文档，则语法在该应用所有加载的文档中都是激活的。在<choice>元素中的<Grammar>元素不能指定它的 Scope 属性，否则抛出一个 error.badfetch 事件。

有时候，Form 可能需要一些语法在整个文档中都激活，而另一些语法只有在该 Form 中激活，这样做的目的是最大限度地减少语法叠交的问题。如果<Grammar>元素的 Scope 属性值和 Form 的 Scope 属性值不同，应该单独指定自己的 Scope 属性值。

8.3.4 语法激活

如果一个输入匹配了多个激活的语法，则语法的优先权按照以下顺序定义：

- 该 Input Item 的语法，包括该 Input Item 包含的<Link>元素中的语法。
- 该 Form 的语法，包括该 Form 包含的<Link>元素中的语法。
- 该文档包含的<Link>元素中的语法，及该文档的<Menu>元素和<Form>元素中 Scope 属性为 Document 的<Grammar>元素的语法。
- 该应用根文档包含的<Link>元素中的语法及该应用根文档的<Menu>元素和<Form>元素中 Scope 属性为 Document 的<Grammar>元素的语法。
- 平台定义的默认事件处理的语法，例如 Help、Exit 和 Cancel。

如果一个输入匹配了多个激活的同一优先权的语法，则语法的优先权由文档顺序决定。当平台期待一个输入却没有任何语法激活时，平台应抛出一个 `error.semantic` 事件。

如果 Form 元素的 Modal 属性为 True，则在等待输入的时候，除了该元素的语法外，其他的语法都是失效的。如果该输入匹配了另外一个 Form 或 Menu 中的语法，而不是当前 Form 或 Menu，则控制交给那个被匹配的语法所在的 Form 或 Menu，且当前 Form 或 Menu 中的数据丢失。

8.4 系统输出

8.4.1 <Prompt>元素

在 VoiceXML 程序中，系统输出由<Prompt>元素实现，<Prompt>元素用于控制合成语音和预录制音频的输出。从概念上讲，提示语是即时排队播放的，因此解释器会一直播放提示语，除非需要用户提供输入。从这一点看，提示语被播放，同时系统也在等待用户输入。一旦收到来自语音识别（或 DTMF 识别）子系统的输入，解释器就会继续执行。

<Prompt>元素的属性见表 21。

表 21 <Prompt>元素的属性

Bargein	控制用户是否可以打断该提示语，值为 True 或 False。当为 True 时，用户可打断系统提示。默认为 Bargein Property 的值
Bargeintype	设置 Bargein 的类型，值为‘speech’或者‘hotword’，默认为 Bargeintype Property 属性值
Cond	ECMAScript 表达式。只有当其结果值为 boolean true，该<Prompt>元素才被执行，否则不执行。缺省时为 True
Count	它是一个数字，用于标识不同的<Prompt>，默认值为“1”
Timeout	等待随后用户输入的时间,可以由 ms 或者 s 描。Timeout 属性决定了接下来用户输入的 Noinput Timeout
Xml:lang	该<Prompt>的语言标识符。它默认为该文档的"xml:lang"属性的值
Xml:base	声明一个基本 URI，用于解析该<Prompt>元素中的相对 URI。<Prompt>元素中的该属性的优先权高于<Vxml>元素。如果缺省，它的值从文档层次继承得到

<Prompt>中的子元素由 W3C SSML 规范定义，见表 22。

表 22 <Prompt>中的子元素

元 素	作 用
<Audio>	指定要播放的音频文件和文本
<Break>	在语音输出中插入一个停顿
<Desc>	为<Audio>元素中的非语音音频提供一段描述
<Emphasis>	在所包含的文本中加强重音
<Lexicon>	为相应的提示语指定发音词典
<Mark>	VoiceXML 平台忽略该元素
<Metadata>	指定该提示语的 XML 元数据内容
<Paragraph>或<p>	标识一个段落，可包括 0 个或多个句子
<Phoneme>	为包含的文本指定发音
<Prosody>	包含在一个规则定义中的元素，该元素提供了一个匹配该规则的输入的例子
<Say-as>	为包含的文本指定结构的类型
<Sentence>或<s>	把所包含的文本标识为一个句子
<Sub>	用指定的文本的发音代替所包含的文本的发音
<Voice>	指定文本的声音特性

VoiceXML 平台应是 SSML 中定义的 ConForming Speech Synthesis Markup Language Processor。虽然平台要处理文档中定义的一个或多个“xml:lang”属性，但是并不要求平台应是支持多语言的。当遇到一种不支持的语言时，平台要抛出 error.unsupported.language 事件，该事件在它的“_message”变量中指定了这种不支持的语言。

8.4.2 <Prompt> Barge-in

如果执行平台支持 Bargein，则开发者就能够指定用户是否可以使用 DTMF 或语音打断正在播放的提示语。这样可以加快交互的速度。并不是所有的情况我们都希望用户可以打断提示语。如果要求用户应听完所有的警告，合法的通知或广告，则应该让 Bargein 失效，可通过设置 Bargein 属性达到。

如果一个<Prompt>元素的 Bargein 属性值为 True，则用户可以打断该提示语；反之，如果 Bargein 属性值为 False，则用户应听完该<Prompt>元素的提示语。如果有几个<Prompt>元素排队，则遵循正在播放的那个<Prompt>元素的 Bargein 属性值。如果在该序列中的某个<Prompt>元素出现 Bargein，则其后的<Prompt>元素都不会被播放（即使其中有的<Prompt>元素的 Bargein 属性值为 False）。如果没有指定 Bargein 属性，则使用 Bargein Property 的值。

当 Bargein 属性值为 False，则在播放<Prompt>元素提示语时，输入是会被缓冲的，且在跳转状态中，所有的 DTMF 输入缓冲均被删除。

不是所有的语音识别引擎或执行平台都支持 Bargein。一个支持 Bargein 的平台应支持以下中描述的 Bargeintype 中的一种。

表 23 Barge-in 类型

Speech	当检测到有 DTMF 或语音输入时，提示语马上停止播放，不管输入是否匹配激活的语法
Hotword	直到检测到输入完全匹配一个激活的语法时，才马上停止播放提示语。不匹配语法的输入被忽略，且不会产生 Nomatch 的事件

8.4.3 分级提示

分级提示即每次播放的提示语都可以不一样。当用户对服务更熟悉时，提示语可变得更简短；而用户需要更多的帮助时，提示语可变得更详细；或者提示语不停的改变只是为了使交互更有趣。

每个 Input Item、<Initial>元素和<Menu>元素都有一个内部的提示语计数器，当进入该<Form>元素或<Menu>元素时，这些计数器被置为 1。无论系统何时播放一个提示语，它相应的计数器都会增加 1。这就是支持分级提示的机制。

8.5 事件处理

8.5.1 事件

当用户没有响应，或没有清楚的响应或请求帮助等，平台会抛出事件。如果发现 VoiceXML 文档中有语义错误时，或执行了<Throw>元素时，解释器也会抛出事件。事件由字符串表示。

每个可能发生事件的元素都可以通过以下方式进行事件处理：

- <Catch>
- <Error>
- <Help>
- <Noinput>
- <Nomatch>

如果需要的话，元素会从它的上层元素直接继承<Catch>元素。例如，<Field>元素中没有用<Nomatch>

去处理 Nomatch（语音识别失败）事件，但是包含<Field>的<Form>元素中有<Nomatch>，则使用该<Form>元素中的<Nomatch>元素进行相关的处理。这样，可以在任何层次指定公共的事件处理，该事件处理在该层次所包含的所有元素中都可以使用。

8.5.2 事件抛出

<Throw>元素用于抛出一个事件，可以抛出预定义或自定义的事件。

<Throw>元素的属性见表 24。

表 24 <Throw>元素的属性

Event	要抛出的事件
Eventexpr	ECMAScript 表达式，它的结果值为要抛出的事件
Message	有关被抛出的事件的信息。对于抛出的预定义的事件，该 Message 的值是依赖于平台的。在<Catch>元素中，可通过“_message”变量取得该信息
Messageexpr	ECMAScript 表达式。其结果值为有关被抛出的事件的信息

应正确的指定属性“Event”、“Eventexpr”或“Namelist”中的一个，否则会抛出一个 error.badfetch 事件。可以正确的指定属性“Message”或“Messageexpr”中的一个，否则会抛出一个 error.badfetch 事件。

8.5.3 事件捕获

<Catch>元素将文档、对话、Form Item 与事件捕获联系起来，它包含了可执行的内容。

<Catch>元素的匿名变量包括一个专有的变量“_event”，它保存了被捕获的事件名。

<Catch>元素的匿名变量也包括专有变量“_message”，它保存了来自相应的<Throw>元素的 Message 字符串的值，或者平台为预定义事件产生的依赖于平台的值。如果被抛出的事件没有指定 Message，则该“_message”的值为 ECMAScript 的 Undefined。

如果<Catch>元素包含了一个<Throw>，且<Catch>要捕获的事件和<Throw>元素抛出的事件一样的话，会形成一个无限循环，平台要能够检测到这种情况，并抛出一个语义错误。

<Catch>元素的属性见表 25。

表 25 <Catch>元素的属性

Event	要捕捉的一个或多个事件。可以指定多个事件的列表，事件之间用空格隔开。这样表示该<Catch>元素将捕捉列表中所有的事件。在这种情况下，每个事件有一个单独的计数器。如果没有指定该属性，则表示所有的事件都会被捕捉
Count	表示事件发生的次数（默认为 1）。该属性可以让你对同一事件发生的不同次数进行不同的处理。每个<Form>，<Menu>和 Form Item 对发生的每个事件都有一个事件计数器。Item 级别的事件计数器用于在访问该 Form Item 和执行该 Form Item 的<Filled>元素时抛出的事件。Form 级别和 Menu 级别的事件计数器用于在初始化该 Dialog 和执行 Form 级别的<Filled>元素时抛出的事件。当<Menu>元素和<Form>元素被重新进入时，该 Form 级别或 Menu 级别的事件计数器被重置。<Clear>元素不会重置该 Form 级别或 Menu 级别的事件计数器。 每当<Form>元素被重新进入时，该<Form>元素所包含的 Item 的 Item 级别的事件计数器被重置。当 Item 被<Clear>元素重置时，该 Item 的事件计数器也被重置。如果没有离开该 Form，且该 Item 被重新进入时，该 Item 的事件计数器不会被重置。 事件计数器增加时，它的全名和匹配该事件的各个前缀都会增加。例如，事件“event.foo.1”的发生使“event.foo.1”、“event.foo”和“event”的事件计数器都增加 1。
Cond	ECMAScript 表达式。只有当其结果值为 Boolean True，才执行该元素。缺省时为 True

8.5.4 快捷标记

<Error>、<Help>、<Noinput>和<Nomatch>元素都是<Catch>元素特定形式的快捷标记，这些元素见表 26。

表 26 <Error>、<Help>、<Noinput>和<Nomatch>元素具有的属性

Count	要捕获的事件发生的次数（和<Catch>元素中的 Count 属性一样）
Cond	一个表达式，用于测试该元素是否捕捉该事件（和<Catch>元素中的 Cond 属性一样），默认为 True

8.5.5 Catch 元素的选择

在同一个 VoiceXML 应用中，可能存在多个 Catch 元素。当一个事件被抛出时，应检查该事件被处理的作用域及包含该作用域的所有作用域，根据以下的算法以确定最合适的<Catch>元素：

- 将当前作用域内和包含该作用域的所有作用域内的所有的<Catch>元素形成一个列表，先以作用域排序（由当前作用域开始），在每个作用域内再以文档顺序排序。
- 去掉该列表中于当前抛出的事件不匹配的元素；去掉 Cond 属性值为 Boolean False 的元素。
- 找出“正确的计数”：在该列表的<Catch>元素中 Count 属性值最高的，且小于或等于当前的 Count 值的<Catch>元素。
- 选定该列表中计数正确的第一个<Catch>元素。
- 如果抛出的事件名匹配了<Catch>元素中的事件名，则这是一个精确匹配，或是前缀匹配，或者该<Catch>元素没有指定 Event 属性。如果<Catch>元素 Event 属性值中的事件是所抛出的事件的前缀，则为前缀匹配。点号是前缀的分隔符，点号后面的字符都去掉。空的字符串可以匹配任何事件。

在<Catch>元素的选定算法中，文档顺序靠前的<Catch>元素的优先权比文档顺序靠后的高。该算法并没有规定，指定的事件越明确的<Catch>元素的优先权越高，因此在一般情况下，我们建议，指定的事件越明确的<Catch>元素，其文档顺序应越靠前；而越不明确的<Catch>元素，其文档顺序应越靠后。一个元素如果需要的话，将从它的每个上层元素直接继承<Catch>元素。

8.5.6 事件默认处理与预定义事件

如果开发者没有指定对 Noinput、Help、Nomatch、Cancel、Exit 和 Error 这几个事件的事件处理，解释器应该隐式的提供这些事件默认的事件处理。

系统对各种事件和错误默认的事件处理可总结为以下两点。

- 是否提供音频作为响应；
- 执行什么操作。如果提供音频作为响应，收集播放的内容是依赖于平台的。

表 27 Noinput、Help、Nomatch、Cancel、Exit 和 Error 事件处理

事件类型	是否提供音频	执行的操作
Cancel	否	没有 Reprompt
Error	是	退出解释器
Exit	否	退出解释器
Help	是	Reprompt
Noinput	否	Reprompt
Nomatch	是	Reprompt
Maxspeechtimeout	是	Reprompt
All others	是	退出解释器

除了以上几个事件，VoiceXML 还设置了以下预定义事件。

1) error.badfetch

当解释器环境已经解释到某个文档中需要获取一个文档的地方，且获取该文档失败，则解释器环境会抛出该事件。不支持的 Scheme 应用，不对的 URI，客户端退出，信息传送错误，Timeout，安全违规，不支持的资源类型，资源类型不匹配，文档解析错误和 Scheme 指定的各种错误代码的返回都会导致获取失败。

如果解释器环境根据环境预取了一个文档，但是后来并不需要这个文档，则不会抛出 error.badfetch 事件。类似的，如果获取一个音频文件失败，但是该音频有候补的音频或有候补的文本，且获取候补的音频或文本成功，则不会抛出 error.badfetch 事件。

当解释器环境将要跳转到一个新的文档时，一直准备抛出 error.badfetch 事件，直到它能够执行这个新的文档为止，但是，如果抛出事件，也只是在需要该新文档的时候才抛出。在需要该新文档之前是不会抛出的。执行这个新文档是否包括变量初始化是依赖于平台的。

2) error.badfetch.http.<response code>

注：万维网联盟（W3C）制定 Voice XML 中就只预留了一个参数的名称。

3) error.badfetch.protocol.<response code>

当获取失败时，解释器环境应用一个具体的事件类型表明是遇到了哪一个特定的 HTTP 或其他的协议的响应码（Response Code），IETF RFC2616 中定义了 HTTP 响应码的值。这样对于一个获取失败的文档，应用就可以进行不同的处理。其他协议的响应码的值是依赖于相应的协议的。

4) error.semantic

当在 VoiceXML 文档中发现运行时（Run-TIME）错误时，抛出该事件。例如取子串越界错误或引用 Undefined 的变量。

5) error.noauthorization

当应用试图执行一个平台没有授权的操作时，抛出这个事件。例如通过依赖于平台的 Object 访问受保护的数据库，或不适当的进入内置语法等。

6) error.noresource

这是一个运行时错误。在执行期间，如果请求的平台资源不可用时，抛出这个事件。

7) error.unsupported.builtin

当平台不支持所请求的内置类型/语法时，抛出这个事件。

8) error.unsupported.Format

当平台不支持所请求的资源的格式时，抛出这个事件。例如一个不支持的语法格式或媒体格式。

9) error.unsupported.language

当平台不支持语音合成或语音识别的语言时，抛出这个事件。

10) error.unsupported.element

当平台不支持给定的元素（该元素是这个技术规范中定义的 VoiceXML 元素）时，抛出这个事件。例如，如果平台没有实现<Script>元素，在使用该元素时，应抛出 error.unsupported.script 事件。这样可以

让开发者使用事件处理来适应不同的平台。

8.6 可执行内容

8.6.1 可执行内容及其元素处理

可执行内容指的是一段程序逻辑，例如出现在以下元素中。

- <Block>元素中；
- <Filled>元素中；
- 事件处理中（<Catch>、<Help>元素等）。

可执行元素按照它们的文档顺序被依次执行。如果某个可执行元素产生一个错误，则这个错误马上被抛出。其后的可执行元素就不会被执行了。

这一节描述的是所有可以出现在可执行内容中的元素。

8.6.2 <var>

<var>元素用于声明一个变量。

VoiceXML 变量和 ECMAScript 变量是完全等同的：它们在同一个变量空间。VoiceXML 变量可在<Script>元素中使用，就像<Script>元素中定义的变量可在 VoiceXML 中使用一样。使用<Var>元素声明一个变量和在<Script>元素中用 var 语句声明的变量效果是等同的。<Script>元素可出现在任何<Var>元素能够出现的地方。Form Item 也可以声明 VoiceXML 变量。

VoiceXML 中变量的命名规则与在 ECMAScript 中一样，但是以下划线 “_” 开头或以美元符号 “\$” 结尾命名的变量被保留用于内部使用。VoiceXML 变量，包括 Form Item 变量，一定不能包含 ECMAScript 的保留字。在有关的正确性上应遵循 ECMAScript 的规则。例如，变量名应是唯一的，且声明变量一定不能够包括点号，即禁止 “var x.y” 在 ECMAScript 中是非法的声明。如果变量名和命名规则或 ECMAScript 规则冲突，抛出 ‘error.semantic’ 事件。

变量的几个作用域见表 28。

表 28 变量的作用域

Session	这是一些只读的变量，在整个用户会话期间都是可用的。它们由解释器环境声明并设置。在 VoiceXML 文档中不能再声明新的 Session 变量
Application	这是一些由<Var>元素和<Script>元素声明的变量，且<Var>元素和<Script>元素都是应用根文档的<Vxml>元素的子元素。这些变量在加载应用根文档时被初始化。在应用根文档被加期间，这些变量都是存在的，且在应用根文档和应用叶文档中都可以访问它们。注意，当在应用根文档中执行时，Document.x 和 Application.x 是等价的
Document	这是一些由<Var>元素和<Script>元素声明的变量，且该<Var>元素和<Script>元素都是该文档的<VXML>元素的子元素。这些变量在加载文档时被初始化。在文档被加期间，这些变量都是存在的，且只能在该文档中访问它们。除非该文档是应用根文档，在这种情况下，在应用根文档和应用叶文档中都可以访问它们
Dialog	每个 Dialog（<Form>元素或<menu>元素）都有一个 Dialog 作用域，当用户访问该 Dialog 时，该作用域才存在。Dialog 变量是由<Var>元素和<Script>元素声明的变量，且该<Var>元素和<Script>元素都是<Form>元素或<Menu>元素的子元素；Dialog 变量也可以是由各种 Form Item 声明的变量。<Form>元素的<Var>子元素和<Script>子元素在该<Form>元素被首次访问时初始化。<Var>元素里面的可执行内容在执行可执行内容时被初始化。Form Item 变量在该 Form Item 被选定时初始化
Anonymous	每个<Block>元素，<Filled>元素和<Catch>元素定义了一个新的 Anonymous 作用域。在这些元素中声明的变量的作用域即为 Anonymous。它们只能在相应的<Block>元素，<Filled>元素或<Catch>元素中被引用

<Var>可以出现在任何可执行的内容中，也可以作为<Form>元素或<Vxml>元素的子元素出现。

如果它出现在可执行的内容中，它声明的变量作用域为 Anonymous，且作用在包含它的<Block>，

<Filled>或<Catch>元素中。在这种情况下，只有<var>元素被执行的时候才声明该变量。如果该变量在这个作用域已经声明了，则其后的声明将被看作是赋值，就像在 ECMAScript 中一样。

如果<Var>元素作为<Form>元素的子元素出现，则它声明的变量的作用域为该 Form 的 Dialog。在这种情况下，在该 Form 的初始化阶段就进行变量声明。

如果<Var>元素作为<Vxml>元素的子元素出现，它声明的变量的作用域为 Document；如果该文档是应用根文档，它声明的变量的作用域为 Application。当文档被初始化的时候就进行变量声明，且初始化是按照文档的顺序进行的。

<Var>的属性见表 29。

表 29 <Var>的属性

Name	保存变量值的变量名
Expr	该变量的初始值（可选的）。如果没有指定该属性，该变量保持它当前的值（如果它当前有值的话）；如果没有给定初始值，变量初始值默认为 ECMAScript 的 Undefined

8.6.3 <Assign>

<Assign>元素用来给变量赋值。

给没有用<var>元素或在<Script>元素中的 var 语句显式声明的变量赋值是非法的。试图给一个没有声明的变量赋值，平台会抛出 error.semantic 事件。

<Assign>元素的属性见表 30。

表 30 <Assign>元素的属性

Name	要被赋值的变量名
Expr	要赋给该变量的值

8.6.4 <Clear>

<clear>元素的作用是重置（清零）一个或多个变量，包括 Form Item。重置 Form Item 包括以下的操作：

- 把该 Form Item 变量置为 ECMAScript 的 Undefined。
- 重新初始化该 Form Item 的提示语计数器和事件计数器。

<Clear>的属性见表 31。

表 31 <Clear>的属性

Namelist	要重置的变量清单，它除了可以包括 Form Item 也可以包括变量名。如果没有指定，当前 Form 的所有 Form Item 都被重置
----------	---

8.6.5 IF, Elseif, Else

<If>元素用于判断条件逻辑，它可以和<Elseif>、<Else>搭配使用。

8.6.6 <Prompt>

提示语可以采用<Prompt>元素的一般形式出现在可执行内容中，除非不使用<Prompt>元素的 Count 属性。此外，它的 Cond 属性也可用在可执行内容中。提示语可以被封装在<Prompt>和</Prompt>中，或使用 PCDATA 表示。只要是<Prompt>元素允许出现的地方，使用 PCDATA xyz 等价于使用<Prompt>xyz</Prompt>。

8.6.7 <Reprompt>

<Reprompt>用于重新播放提示语，只用于<Catch>元素内。

8.6.8 <Goto>

<Goto>元素用于：

- 跳转到当前 Form 的另一个 Form Item；
- 跳转到当前文档的另一个 Dialog；
- 跳转到另一个文档。

要跳转到另一个 Form Item，使用 Next 属性，或使用 Expritem 属性，如果该 Form Item 名可以通过 ECMAScript 表达式计算得到的话。

要跳转到同一个文档的另一个 Dialog，使用 Next 属性，或使用 expr 属性指定一个 URI。

要跳转到另一个文档使用 next 属性，或使用 Expr 属性指定一个 URI。

URI 可以是当前文档的绝对或相对的 URI。要指定下一个文档开始的 Dialog，可以使用对应于该 Dialog 的 ID 属性值的 URI 段。如果没有指定 URI 段，则以要跳转到的文档的第一个 Dialog 为开始的 Dialog。

跳转到当前文档的另一个 Dialog 会导致旧的 Dialog 变量丢失，即使是一个 Dialog 跳转到它自己也会丢失。同样的，使用相对或绝对的 URI 跳转到另一个文档，也会丢失文档变量，即使是一个文档跳转到它自己也会丢失。然而，当跳转到一个带有段标识符的空的 URI 引用时，文档变量不会丢失。

<Goto>元素的属性见表 32。

表 32 <Goto>元素的属性

Next	要跳转到的 URI
Expr	一个 ECMAScript 表达式，其结果值为要跳转到的 URI
Nextitem	当前 Form 中下一个要访问的 Form Item 名
Expritem	一个 ECMAScript 表达式，其结果值为当前 Form 中下一个要访问的 Form Item 名
Fetchaudio	它默认为 Fetchaudio <Property>
Fetchhint	它默认为 Documentfetchhint <Property>
Fetchtimeout	它默认为 Fetchtimeout <Property>
Maxage	它默认为 Documentmaxage <Property>
Maxstale	它默认为 Documentmaxstale <Property>

应正确地指定属性“Next”、“Expr”、“Event”和“Eventexpr”中的一个，否则会抛出 error.badfetch 事件。

8.6.9 <Submit>

<Submit>元素用于提交信息给 Web 服务器，然后跳转到 Web 服务器响应后返回的文档。和<Goto>元素不一样的是，<Submit>元素可以通过 HTTP GET 或 POST 请求提交一个变量列表到文档服务器。

<Submit>元素的 Next 或 Expr 属性指定了要跳转到的 Dialog。该 URI 每次都要被获取，即使它只包含了一个段。

<Submit>元素的属性见表 33。

表 33 <Submit>元素的属性

Next	要引用的 URI
Expr	ECMAScript 表达式，它的结果值为要引用的 URI
Namelist	要提交的变量名列表。默认是提交所有有命名的 Input Item。如果指定了 Namelist 属性，它可以包含一个或多个变量名，以空格隔开。在 VoiceXML 和 ECMAScript 中声明的变量都能被提交。
Method	请求的方法：Get（默认的）或 Post
Enctype	被提交的文档的媒体编码类型，默认为 application/x-www-form-urlencoded。解释器也应支持 multipart/form-data，也可以支持其他的编码类型

表 33 (续)

Fetchaudio	它默认为 Fetchaudio <Property>
Fetchhint	它默认为 Documentfetchhint <Property>
Fetchtimeout	它默认为 Fetchtimeout <Property>
Maxage	它默认为 Documentmaxage <Property>
Maxstale	它默认为 Documentmaxstale <Property>

应正确的指定属性 Next 或 Expr 中的一个，否则会抛出 error.badfetch 事件。

8.6.10 <Exit>

<Exit>把控制返回给解释器环境，由解释器环境决定下一步做什么。

该元素和<Return>元素的不同之处在于它终止了所有加载的文档，而<Return>则是从一个<Subdialog>调用中返回。如果<Subdialog>调用了一个新的文档（或应用），则<Return>会终止被调用的文档，控制返回到<Subdialog>元素继续执行。

一旦<Exit>把控制返回给解释器环境，解释器环境就可以做它想做的操作。例如，它可以给用户播放上一级<Menu>元素中的提示语，或者挂机，或者把用户转接到人工坐席。

它的属性见表 34。

表 34 <Exit>元素的属性

Expr	一个 ECMAScript 表达式，该表达式的值就是要返回的值（如“0”，“oops!”或“Field1”）
Namelist	要返回给解释器环境的变量名清单。默认是返回 0 个变量，这样意味着解释器环境将接收到一个空的 ECMAScript 对象

8.6.11 <return>

<Return>元素用于结束一个 Subdialog 的执行，并把控制和数据返回给调用它的 Dialog。

它的属性见表 35。

表 35 <Return>元素的属性

Event	要返回并抛出的事件
Eventexpr	ECMAScript 表达式。其结果值为抛出的事件。它的作用是从 Subdialog 中返回，并抛出该事件
Message	有关被抛出的事件的信息。在<Catch>元素中，可通过“_message”变量取得该信息
Messageexpr	ECMAScript 表达式。其结果值为有关被抛出的事件的信息
Namelist	要返回的变量名。默认为返回 0 个变量，即调用该 subDialog 的 Dialog 将得到一个空的 ECMAScript 对象

当从一个 Subdialog 中返回，一个事件在调用的地方被抛出，或者数据作为一个 ECMAScript 对象被返回，该对象的属性对应于 Subdialog 中的<Return>元素的 Namelist 属性值。如果不是在 Subdialog 中执行，却碰到<Return>元素，则会抛出一个语义错误。

8.6.12 <Script>

<Script>元素允许在 VoiceXML 脚本中使用一段客户端的脚本语言代码，它和 HTML 的<SCRIPT>元素类似。

<Script>元素可以出现在<Vxml>元素和<Form>元素中，或者出现在可执行的内容中（如<Filled>，<if>，<block>，<catch>或<Catch>元素的快捷标记）。在文档被装载后，<Vxml>元素中的<Script>元素和<var>元素都按照它们的文档顺序被依次求值。每次执行到<Form>元素，该 Form 中的<Script>元素和<Var>元素及 Form Item 变量也是按照它们的文档顺序被依次求值。和其他的可执行元素一样，当在可执行的内容中碰到<Script>元素时，都会执行它。

<Script>元素的属性见表 36。

表 36 <Script>元素的属性

| | |
|--------------|--|
| Src | 如果脚本是外部的，则该属性指定了该外部脚本的 URI |
| Charset | 属性 src 指定的脚本的字符编码。平台应支持 ISO/IEC 10646 的 UTF-8 和 UTF-16 编码，也可以支持其他的编码，如 IANA 中定义的。默认为 UTF-8 |
| Fetchhint | 它默认为 Documentfetchhint <Property> |
| Fetchtimeout | 它默认为 Fetchtimeout <Property> |
| Maxage | 它默认为 Documentmaxage <Property> |
| Maxstale | 它默认为 Documentmaxstale <Property> |

VoiceXML 中的<Script>元素没有 Type 属性（这一点和 HTML 的<Script>元素不一样），ECMAScript 是 VoiceXML 必须的脚本语言。

每个<Script>元素在它的父元素的作用域内被执行，即它没有自己的作用域。例如在<Script>元素用 Var 语句声明的变量的作用域为该<Script>元素的父元素（用 ECMAScript 的术语来说，就是“变量对象”为包含<Script>元素的元素的当前作用域）。

8.6.13 <Log>

<Log>元素让应用程序可以产生日志或调试信息，这些信息能够用来帮助开发者开发程序，或进行执行后的程序性能分析。

<Log>元素可以包含文本（CDATA）和<Value>元素的任意组合，产生的信息由文本和<Value>元素的 Cond 属性值串联起来组成。

信息的显示和记录方式是依赖于平台的，Lable 属性的用法也是依赖于平台的。

<Log>元素中的 ECMAScript 表达式根据它们在文档中的顺序求值，使用<Log>元素不应该对文档的解释产生任何的副作用。

<Log>元素的属性见表 37。

表 37 <Log>元素的属性

| | |
|-------|-------------------------------|
| Label | 一个字符串，例如它可以用来表明该<Log>元素的目的 |
| Expr | 一个 ECMAScript 表达式，它的结果值为一个字符串 |

8.7 资源获取与环境信息

8.7.1 资源获取

VoiceXML 解释器环境需要获取 VoiceXML 文档和其他的资源，例如声音文件、语法、脚本和 Object。属性控制着每次和 URI 相关的资源的获取，见表 38。

表 38 控制 URI 相关资源获取的属性

| | |
|--------------|---|
| Fetchtimeout | 等待要获取的资源返回的时间，超过这个时间就抛出一个 Error.badfetch 时间。该值是一个 Time Designation。如果没有指定该属性，则使用最内部的 Fetchtimeout 属性值 |
| Fetchhint | 它定义了解释器环境应该在何时从服务器上获取资源。该属性值为 Prefetch 或 Safe。Prefetch 表示当加载文档时，就可以获取该资源了；而 Safe 表示只有在确实需要获取资源时才去获取资源。如果没有指定该属性，则使用最内部的 Fetchhint 属性值 |
| Maxage | 表示如果该资源存在的时间没有超过指定的时间（以 s 计），则文档可以使用该资源（类似于 HTTP 1.1 即 IETF RFC2616 中的“max-age”）。文档不会使用过期的资源，除非指定了 Maxstale 属性。如果没有指定该属性，则使用最内部的 Maxage 属性值 |
| Maxstale | 表示文档可以使用过期的资源（类似于 HTTP 1.1 即 IETF RFC2616 中的“max-stale”）。如果指定了该属性，则文档会接受过期时间不超过该属性值的资源。如果没有指定该属性，则使用最内部的 Maxstale 属性值 |

表 38 (续)

| | |
|------------|--|
| Fetchaudio | 该 URI 指定了正在获取资源时要播放的音频。如果没有指定, 则使用 Fetchaudio Property 值。如果 Fetchaudio Property 也没有设置, 则在获取资源时不播放任何音频。Audiofetchhint, Audiomaxage, Audiomaxstale Property 控制着该音频的获取, 而 Fetchtimeout Property 则控制资源获取的时间。Fetchaudiodelay Property 控制该音频的播放, Fetchaudiominimum Property 控制该获取音频的时间 |
|------------|--|

当从 URI 获取资源时, Fetchtimeout 属性决定了等待该资源返回的时间 (由需要获取该资源的时候算起), Fetchhint 属性决定了何时获取该资源。VoiceXML 解释器环境的缓存策略采用 Maxage 和 Maxstale 属性。当正在获取下一个文档有明显的延迟时, Fetchaudio 属性就很有用, 它可用于播放背景音乐, 或者公告。当需要的文档已经获取到时, 如果还在播放音频, 则中断播放。在获取 Fetchaudio 属性指定的音频时, 如果出现错误, 则不抛出任何事件, 在获取文档时也不播放任何音频。

8.7.2 资源缓存

和 HTML 的可视浏览器一样, VoiceXML 解释器环境也可以使用缓存, 以提高获取文档和其他资源时的性能。就像在 HTML 网页中有很多图片一样, 在 VoiceXML 文档中也有很多声音剪辑。在可视的浏览器中, 通常都包含有终端用户的控制, 用于刷新那些快要过期的内容。而这在 VoiceXML 解释器中就做不到, 因为它缺少响应的中断用户的控制, 因此, 实现缓存更新是通过在文档中适当使用 Maxage 和 Maxstale 属性判断的。

VoiceXML 解释器环境使用的缓存策略应遵循 HTTP1.1 (见 IETF RFC2616) 中的缓存正确性规则。特别是应遵循到期和缓存控制头, 下面的算法总结了这些规则, 并说明了当请求资源时解释器环境的行为。

- 如果该资源在缓存中不存在, 则使用 Get 方法从服务器获取。
- 如果该资源在缓存中存在,
 - 如果有提供了 Maxage 属性值,
 - 如果缓存中的该资源存在的时间<=Maxage 属性值,
 - 如果该资源已经过期,
 - 则检查 Maxstale 属性值。
 - 否则使用缓存中的该资源。
 - 否则使用 Get 方法从服务器获取相应的资源。
 - 否则,
 - 如果该资源已经过期,
 - 则检查 Maxstale 属性值。
 - 否则使用缓存中的该资源。

检查 Maxstale 属性值是指:

- 如果有提供 Maxstale 属性值,
 - 如果缓存中的该资源已经过期, 且过期的时间没有超过它的 Maxstale 属性值, 则使用缓存中的该资源。
 - 否则使用 Get 方法从服务器获取相应的资源。
- 否则使用 Get 方法从服务器获取相应的资源。

当需要从服务器获取资源时，可以检查一下还在缓存中的文档，“如果该文档已经被更改了，则从服务器获取”。这是一个可行的优化。

如果其他的协议也有 Maxage 和 Maxstale 属性，且直接由 HTTP1.1 支持，则有些资源除了可以由 HTTP 的 URI 定位，也可以由这些协议的 URI 定位。如果该协议不支持资源存在时间（Age）的概念，则解释器环境在接收到该资源时开始计算其存在时间。如果该协议不支持资源失效（Staleness）的概念，则解释器环境应该认为该资源在被接收后马上就过期。

VoiceXML 允许开发者对每个资源（除了<Vxml>元素的 Application 属性引用的文档：对于应用根文档，没有任何机制可以控制它的缓存策略）的每次使用进行缓存策略的控制。

每个和资源相关的元素都可以指定 Maxage 和 Maxstale 属性。给 Maxage 指定一个非 0 的值，可用于获取缓存中还没有过期的最新的资源。如果 Maxage 的属性值为 0，则每次请求都无条件地去获取最新的资源。

Maxstale 属性使开发者可以规定使用不太陈旧的过期的资源（根据 HTTP1.1 的规则）。通过减少获取资源的次数来提高性能。该属性对于那些对服务器端大文件没有过期控制的开发者来讲，特别有用。

8.7.3 资源预取

预取资源是一个可选的特性，它使解释器环境在需要一个资源之前就可以去获取该资源。如果某元素的 Fetchhint 属性值为“Prefetch”，则由该元素确定的资源就可以进行预取。当解释器环境预取资源时，应确保所预取的资源正好是所需要的资源。特别的，如果该 URI 由 Expr 属性确定，在没有给该表达式的变量赋值之前，解释器环境一定不能提前获取资源。同样的，在给 Namelist 属性的变量赋值之前，也不能提前获取<Submit>元素需要的资源。

每次使用资源都应检查该资源是否过期，如果它的 Fetchhint 属性值为“Prefetch”，则预取该资源。检查资源应遵循上节中的缓存策略的规则。

8.7.4 <Meta>与<Metadata>

元数据信息是关于文档的信息，而不是关于文档内容的信息。VoiceXML 提供了两个元素用于表示元数据信息：<Meta>和<Metadata>元素。对于元数据信息，<Metadata>元素比<Meta>元素提供了更全面、更强大的处理。

VoiceXML 没有指定必须的元素据信息，然而，推荐用<Metadata>元素来表示元数据，该元素的信息是 RDF 格式的。

<Meta>元素指定了元信息，有两种用法：

- 第一种是指定整个文档的元数据 Property，它由 Name 和 Content 这对属性表示。
- 第二种是指定了 HTTP 响应头，它由 HTTP-Equiv 和 Content 这对属性表示。

<Meta>的属性见表 39。

表 39 <Meta>元素的属性

| | |
|------------|-------------|
| Name | 该元数据的特性名 |
| Content | 该元数据的特性值 |
| HTTP-Equiv | HTTP 响应头的名称 |

<Metadata>元素是一个容器，所有关于文档的信息都可以使用元数据模式放在该元素中。虽然任何元数据方案都可以和<Metadata>元素一起使用，但是建议使用 RDF 模式中定义的元数据特性。

表 40 <Metadata>元素的属性

| | |
|---------|--|
| Creator | 表示该资源的内容的作者 |
| Rights | 表示该资源相关的版权信息 |
| Subject | 表示该资源内容的主题。一般使用该资源的关键字、主要的短语或类别的代码来表示。建议最好从控制性词汇或正式类别方案中选一个值 |

8.7.5 <Property>

8.7.5.1 作用域与优先级

<Property>元素用于设置系统的特性值。系统的特性影响着平台的运转状态，例如识别方法，Timeout，缓存策略等。

<Property>可以定义在 VoiceXML 应用任何位置，对应不同的作用域：

- 可以在应用根文档中定义<Property>元素，作用于整个应用；
- 可以在<Vxml>级定义，作用于整个文档；
- 可以在<Form>或<Menu>级定义，作用于该 Dialog；
- 可以或在一个 Form Item 中定义，作用于该 Form Item。

<Property>元素作用于它的父元素及该父元素的所有子元素。对于相同的 Property，低级别的优先权高。当在同一级别中给某个 Property 指定了两个不同的值，则以在文档顺序中排后面的为准。在应用根文档中指定的 Property 值是该应用中所有文档的 property 的默认值。在单独的文档中指定的 Property 值的优先权高于在应用根文档中指定的。

如果平台检测到某个 Property 值是非法的，应该抛出 error.semantic 事件。

8.7.5.2 设置依赖于平台的 Property

解释器环境可以任意的提供依赖于平台的 Property。

依赖于平台的 Property 引入了不兼容的因素，降低了平台的可移植性。为了最大限度的降低这些因素，强烈建议如下的方案：

- 依赖于平台的 Property 应使用反转域名命名，以减少潜在的冲突，例如，com.example.foo 和 net.example.foo 就明显的不一样。
- 当解释器环境碰到不支持的 Property 时，宁可忽略这个 Property，一定不能抛出 error.unsupported.Property 事件。

8.7.5.3 设置语音识别相关的属性

语音识别相关的属性见表 41。

表 41 语音识别相关的属性

| | |
|-----------------|--|
| Confidencelevel | 语音识别的 Confidence 级别，它是一个浮点数，取值范围从 0.0 到 1.0。如果 application.lastresult\$.confidence 的值低于该 Property 的值，识别结果会被拒绝（即抛出 Nomatch 事件）。0.0 表示识别需要的最低的 Confidence；1.0 表示最高的 Confidence。该 Property 的值是一个实数标识。默认值为 0.5 |
| Sensitivity | 设置 Sensitivity 级别。值为 1.0 表示对无噪声输入高度敏感，值为 0.0 表示对噪声的最低的敏感。该 Property 的值是一个实数标识。默认值为 0.5 |
| Speedvsaccuracy | 表示在速度和准确性上想要的平衡点。0.0 表示最快的识别速度，1.0 表示最高的准确性。该 Property 的值是一个实数标识。默认值为 0.5 |

表 41（续）

| | |
|-------------------|--|
| Completetimeout | <p>表示在用户语音输入之后和语音识别器得出识别结果(要么接收这个结果,要么抛出 nomatch 事件。)之前需要的静音的 时长。它用于决定语音输入是否为一个激活语法的完全匹配。相反, Incompletetimeout 用于决定语音输入是否为一个激活语法的不完全匹配。</p> <p>长的 Completetimeout 值会延迟识别结果的产生, 并使得计算机的反应变慢。短的 Completetimeout 值可能会导致用户说的话被不适当的结束。合理的 Completetimeout 值一般为 0.3~1.0s。该值是一个时间标识。默认值是依赖于平台的。</p> <p>虽然平台应解析 Completetimeout Property, 但是平台不是应要支持 Completetimeout 的。如果平台选择不支持 Completetimeout, 则应调整 Incompletetimeout Property 的值。如下面描述的</p> |
| Incompletetimeout | <p>识别器得出识别结果后, 跟在用户语音输入后面的需要的静音时长。当静音之前的语音是所有激活的语法的一个不完全匹配时, Incompletetimeout 就起作用了。在这种情况下, 一旦触发了该属性, 则平台不接受该不完全匹配的结果(即抛出一个 Nomatch 事件)。</p> <p>当静音之前的语音是一个激活的语法的完全匹配, 且继续说下去也可能匹配该语法时, Incompletetimeout 也会起作用。相比之下, 当语音是一个激活的语法的完全匹配, 且平台不接受更多的语音时, 使用的是 Completetimeout。</p> <p>太长的 Incompletetimeout 值会延迟识别结果的产生, 且使计算机的响应变慢。太短的 Incompletetimeout 值可能会导致用户说的话被不适当的结束。</p> <p>Incompletetimeout 值通常都比 Completetimeout 值长, 这样才能让用户在说话时可以停顿(例如, 呼吸)。</p> <p>如果平台选择不支持 Completetimeout Property, 则平台应使用 Completetimeout 和 Incompletetimeout 中的最大值作为 Incompletetimeout 的值。该值是一个时间标识</p> |
| Maxspeechtimeout | <p>用户语音输入的最大时长。如果用户语音输入的时间超过该时长, 则抛出一个“Maxspeechtimeout”事件。该值是一个 Time Designation。默认的值是依赖于平台的</p> |

8.7.5.4 设置 DTMF 相关的属性

DTMF 相关的属性见表 42。

表 42 DTMF 相关的 Property

| | |
|-------------------|---|
| Interdigittimeout | 当识别 DTMF 输入时才使用 Interdigittimeout 值。该值是一个时间标识。默认的值是依赖于平台的 |
| Termtimeout | 当识别 DTMF 输入时才使用 Termtimeout 值。该值是一个时间标识。默认的值是“0s” |
| Termchar | DTMF 输入识别时的结束 DTMF 字符。默认值为“#” |

8.7.5.5 设置系统语音合成输出相关的属性

语音合成相关的属性见表 43。

表 43 语音合成相关的 Property

| | |
|-------------|---|
| Bargein | 值为 True 或 False。该属性在播放提示语时起作用。值为 True 时允许 Bargein, 值为 False 时则不允许 Bargein。默认的值 True |
| Bargeintype | 值为 Speech 或 Hotword。该属性用于设置 Bargein 的类型。默认的值是依赖于平台的 |
| Timeout | 当没有输入的时间超过该值时, 平台就抛出一个 Noinput 事件。该值是一个时间标识。默认的值是依赖于平台的 |

8.7.5.6 设置资源获取相关的属性

资源获取相关的属性见表 44。

表 44 资源获取相关的 Property

| | |
|----------------|---|
| Audiofetchhint | 值为 safe 或 prefetch。该属性用于设置平台是否可以预取音频。当值为 Safe 时, 则该音频只有在需要时才去获取, 在需要之前不能进行获取。当值为 Prefetch 时, 则允许平台预取音频, 但不是应要预取音频。默认值为 Prefetch |
| Audiomaxage | 用于设置对于缓存中的音频资源平台可接受的最大的存在时间, 单位为 s。默认的值是依赖于平台的 |

表 44（续）

| | |
|-------------------|--|
| Audiomaxstale | 用于设置对于缓存中已经过期的音频资源平台可接受的最大的失效时间。默认的值是依赖于平台的 |
| Documentfetchhint | 值为 Safe 或 Prefetch。用于设置平台是否可以预取文档。默认值为 Safe |
| Documentmaxage | 用于设置对于缓存中的文档平台可接受的最大的存在时间。单位为 s。默认的值是依赖于平台的 |
| Documentmaxstale | 用于设置对于缓存中的过期文档平台可接受的最大的失效时间，单位为 s。默认的值是依赖于平台的 |
| Grammarfetchhint | 值为 Safe 或 Prefetch。用于设置平台是否可以预取语法。默认值为 Prefetch |
| Grammarmaxage | 用于设置对于缓存中的语法平台可接受的最大的存在时间，单位为 s。默认的值是依赖于平台的 |
| Grammarmaxstale | 用于设置对于缓存中过期的语法平台可接受的最大的失效时间，单位为 s。默认的值是依赖于平台的 |
| Objectfetchhint | 值为 Safe 或 Prefetch。用于设置平台是否可以预取<Object>元素 URI 的内容。默认值为 Prefetch |
| Objectmaxage | 用于设置对于缓存中的 Object 平台可接受的最大的存在时间，单位为 s。默认的值是依赖于平台的 |
| Objectmaxstale | 用于设置对于缓存中过期的 Object 平台可接受的最大的失效时间，单位为 s。默认的值是依赖于平台的 |
| Scriptfetchhint | 值为 safe 或 prefetch。用于设置平台是否可以预取<Script>。默认值为 Prefetch |
| Scriptmaxage | 用于设置对于缓存中的<Script>平台可接受的最大的存在时间，单位为 s。默认的值是依赖于平台的 |
| Scriptmaxstale | 用于设置对于缓存中过期的<Script>平台可接受的最大的失效时间，单位为 s。默认的值是依赖于平台的 |
| Fetchaudio | 在等待获取资源文档或其他资源时要播放的音频的 URI。默认是在获取时不播放任何音频。<Audio>，<Grammar>，<Objects>和<Scripts>元素没有该 Property。音频的获取由 Audiofetchhint，Audiomaxage，Audiomaxstale 控制，Fetchtimeout 则控制获取的时间。音频的播放由 Fetchaudiodelay 控制，Fetchaudiominimum Peoperty 则控制获取的时间 |
| Fetchaudiodelay | 在播放 Fetchaudio 音频之前等待获取延迟的开始的时间。该值是一个时间标识。默认的值是依赖于平台的，如“2s”。当获取的延迟非常短时，不播放任何音频比刚播放一小段音频就被打断好 |
| Fetchaudiominimum | 播放 Fetchaudio 音频最短的时间。一旦开始播放 Fetchaudio 音频，即使资源已经获取到了，如果播放的时间还没有超过该值，也要继续播放。该值是一个时间标识。默认的值是依赖于平台的，如“5s”。一旦用户已经听到 Fetchaudio 音频，则不应该很快就中断播放，这就是该属性的作用 |
| Fetchtimeout | 获取资源 Timeout 的时间，该值是一个时间标识。默认的值是依赖于平台的 |

8.7.5.7 其他属性

其他属性见表 45。

表 45 其他 Property

| | |
|------------|--|
| Inputmodes | 值为 Dtmf 或 Voice。该 Property 用于确定要使用哪种输入方式。在两种方式都支持的平台中，默认值为“Dtmf Voice”。要使语音识别失效，则把该 Property 设置为“Dtmf”，要使 DTMF 识别失效，则把该 Property 设置为“Voice”。该 Property 的一个作用是，在嘈杂的环境中可以使语音识别失效，而只使用 DTMF 识别。在只希望 DTMF 识别时，也可以让语音识别失效。该 Property 不能控制语法的激活。例如，当 Inputmodes 为“Dtmf”时，语音语法也可以被激活。然而，这些语音语法不会被匹配，因为语音输入方式没有激活 |
| Universals | 平台可以可选的提供一些依赖于平台的通用的语法（Universal Command Grammars），例如“Help”，“Cancel”或“Exit”语法，这些语法总是激活的，且会产生相应的事件。
一些应用通常需要定义它们自己的通用的语法，以提高应用的可移植性或提供一个与众不同的界面。他们通过<Link>元素指定新的通用语法。然后用该 Property 避开默认的语法。该 Property 不会影响默认的<Catch>的事件处理。
该 Property 的默认值为“None”，它表示所有平台默认的通用语法都失效。若该 Property 的值为“All”，则表示所有平台默认的通用语法都激活。要激活单个的语法，可以把该语法放在列表中，各个语法以空格隔开。例如，“Cancel Exit Help” |
| Maxnbest | 该 Property 用于控制数组“application.lastresult\$”的最大长度。该数组的长度不能超过 Maxnbest 的值。该 Property 的最小值为 1。默认值为 1 |

8.7.6 <Param>

<Param>元素用于指定要传递给 Subdialog 或 Object 的值。它是以 HTML 的<Param>元素为模型的。它的属性见表 46。

表 46 <Param>元素的属性

| | |
|-----------|--|
| Name | Object 或 Subdialog 调用时的参数名 |
| Expr | ECMAScript 表达式，它把它的结果值赋给相应的 Name 属性指定的参数 |
| Value | Name 属性指定的参数的值 |
| Valuetype | 它的值为 Data 或 Ref，默认为 Ref。在 Object 中，它用于表明 Name 属性的值是数据还是 URI（Ref）。该属性不用在<Subdialog>元素中，因为<Subdialog>元素中的参数值都是数据 |
| Type | 它表示当 Valuetype 的值为 Ref 时，Name 属性指定的 URI 的媒体类型（Media Type）。该属性只用于<Object>元素中的<Param>元素 |

应正确地指定属性 Expr 或 Value 中的一个，否则会抛出 Error.badfetch 事件。

虽然特定的 Object 可能需要指定 Valuetype 和 Type 的属性值，但是通常这两个属性是可选的。当 <Param>元素包含在<SubDialog>元素中时，<Param>元素指定的值用于初始化 SubDialog 调用的 Dialog 的<Var>元素。当<Param>元素包含在<Object>元素中时，参数的用法是依赖于所调用的 Object 的，它不属于本规范说明书的范围。

8.7.7 值标识规范

一些 VoiceXML 参数的值遵循在 W3C CSS（Cascading Style Sheet）中使用的约定。

实数和整数只能在十进制中使用。整数由一个或多个“0”~“9”组成。实数可以是一个整数，也可以是 0 或小数。实数和整数在前面加一个“+”或“-”表示正负。

时间标识由一个非负的实数和时间单位标识符组成。时间的单位标识符有以下几种。

- ms：毫秒；
- s：秒，例如：“3s”，“850ms”，“0.7s”，“-5s”或“+1.5s”。

9 语音浏览器与语音服务器信息交互格式

语音浏览器与语音服务器的信息交互格式需依照 IETF RFC4463（MRCP）。在交互过程中，语音浏览器可以视作支持 MCRP 协议的客户端，语音服务器可以视作支持 MRCP 协议的服务器。

9.1 MRCP 消息格式

9.1.1 请求 Request 格式

MRCP 请求的通用格式为：

```
Request-line = method-name SP request-id SP mrcp-version CRLF
SP = %x20
CRLF = CR LF
method-name = synthesizer-method / recognizer-method
request-id = 1*DIGIT
mrcp-version = "MRCP" "/" 1*DIGIT "." 1*DIGIT
```

以上格式中的规定如下：

- ◆ Method-name 标识不同的语音合成/识别方法调用；

- ◆ Request-ID 是唯一数字串，用于对不同的请求进行标记；
- ◆ Mrcp-version 标识 MRCP 版本名，通常为 MRCP 1.0。

9.1.2 响应 Response 的格式：

MRCP 响应的通用格式为：

Response-line = mrcp-version SP request-id SP status-code SP request-state CRLF

Request-state = "COMPLETE" / "IN-PROGRESS" / "PENDING" ;

以上格式的规定如下：

- Request-ID 部分应与其对应的请求的 request-id 保持一致；
- Status-Code 返回状态码；
- Request-State 标识完成、处理中、队列中 3 个状态，后两个状态意味着服务器还将发送异步事件

到客户端。

9.1.3 事件 Event 的格式：

MRCP 事件的通用格式为：

Event-line = event-name SP request-id SP request-state SP mrcp-version CRLF

event-name = synthesizer-event / recognizer-event

以上格式的规定如下：

- Event-Name 标识语音合成/语音识别事件；
- Request-ID 部分应与其对应的请求的 Request-ID 保持一致；
- Request-State 标识完成、处理中、队列中 3 个状态，后两个状态意味着服务器还将发送异步事件

到客户端；

- Mrcp-Version 标识 MRCP 版本名，通常为 MRCP 1.0。

9.1.4 消息头 Message Header 格式：

MRCP 消息的消息头格式为：

message-header = 1* (generic-header / resource-header)

generic-header = active-request-id-list

/ proxy-sync-id

/ content-id

/ content-type

/ content-length

/ content-base

/ content-location

/ content-encoding

/ cache-control

/ logging-tag

resource-header = synthesizer-header / recognizer-header

Active-request-id-list = "Active-Request-Id-List" ":" request-id* ("," request-id) CRLF

以上格式的规定如下。

- Active-request-id-list: 已激活的请求队列;
- Proxy-sync-id: 当 Barge-in-Occurred 被调用, 识别服务与语音合成服务分布在不同服务器时, Proxy-sync-id 作为 Barge-in 的标记, 由识别服务器发出, 经 MRCP 客户端转发到语音合成服务器, 用于信息同步。
- Content-id: 内容标记符, 依据 IETF RFC2392, IETF RFC2822, IETF RFC2046 定义, 表示 MRCP 信息所在的特定会话内容, 例如 session:URI;
- Content-type: 内容类型, 描述 MRCP 信息包含的语音标记、语法、识别结果的表达形式, 如果是多种类型混在在一起, 可以使用 Multi-part/mixed 描述;
- Content-length 内容长度, 标记 MRCP 信息体的长度;
- Content-base 内容基准, 当在 MRCP 信息中使用了相对路径时, 可以使用 Content-base 标识基准路径;
- Content-location;
- Content-encoding: 内容编码方式;
- Cache-control: 当 MRCP 信息中涉及媒体信息的存取, Cache-control 可以指定缓存原则;
- Logging-tag: 日志标记符, 用于在系统日志中标记该 MRCP 信息, 用于问题定位等用途;
- Resource-header: 语音合成 / 语音识别信息头。

9.2 语音合成消息头、方法、事件

9.2.1 语音合成消息头

语音合成消息头的格式及其参数如下:

```
synthesizer-header = jump-target      ; 快进或者回退的跳转长度
                    / kill-on-barge-in ; 合成输出因语音或者 DTMF 中断而停止
                    / speaker-profile  ; 语音合成发音折信息
                    / completion-cause ; 结束原因
                    / voice-parameter  ; 声音特性参数, 例如性别, 年龄等
                    / prosody-parameter ; 韵律特性参数, 例如音量, 语速等
                    / vendor-specific  ; 自定义参数
                    / speech-marker    ; 语音标记, 用于合成声音跳转
                    / speech-language ; 语音合成语言设置
                    / fetch-hint       ; 外部资源获取方式
                    / audio-fetch-hint ; 外部声音资源获取方式
                    / fetch-timeout    ; 获取操作超时设置
                    / failed-uri       ; 导致获取操作失败的目标路径
                    / failed-uri-cause ; 导致获取操作失败的原因
                    / speak-restart   ; 重新开始语音合成输出
                    / speak-length    ; 语音合成长度
```

9.2.2 语音合成方法

语音合成方法的格式及其参数如下:

synthesizer-method = "SET-PARAMS" ; 参数设置
/ "GET-PARAMS"; 参数获取
/ "SPEAK" ; 开始合成并声音输出
/ "STOP" ; 退出合成与声音输出
/ "PAUSE" ; 暂停合成与声音输出
/ "RESUME" ; 继续合成与声音输出
/ "BARGE-IN-OCCURRED"; 发生语音或者 DTMF 中断
/ "CONTROL" ; 语音合成控制

9.2.3 语音合成事件

语音合成事件的格式及其参数如下:

synthesizer-event = "SPEECH-MARKER"; 合成文本位置标记
/ "SPEAK-COMPLETE"; 语音合成输出完成

9.2.4 语音合成消息头与事件、方法的关系

语音合成消息头与事件、方法的关系见表 47。

表 47 语音合成消息头与事件、方法的关系

| 信息头 | 是否支持 | 方法 / 事件 / 响应 |
|-------------------|-----------|--|
| Jump-target | MANDATORY | SPEAK, CONTROL |
| Logging-tag | MANDATORY | SET-PARAMS, GET-PARAMS |
| Kill-on-barge-in | MANDATORY | SPEAK |
| Speaker-profile | OPTIONAL | SET-PARAMS, GET-PARAMS, SPEAK, CONTROL |
| Completion-cause | MANDATORY | SPEAK-COMPLETE |
| Voice-parameter | MANDATORY | SET-PARAMS, GET-PARAMS, SPEAK, CONTROL |
| Prosody-parameter | MANDATORY | SET-PARAMS, GET-PARAMS, SPEAK, CONTROL |
| Vendor-specific | MANDATORY | SET-PARAMS, GET-PARAMS |
| Speech-marker | MANDATORY | SPEECH-MARKER |
| Speech-language | MANDATORY | SET-PARAMS, GET-PARAMS, SPEAK |
| Fetch-hint | MANDATORY | SET-PARAMS, GET-PARAMS, SPEAK |
| Audio-fetch-hint | MANDATORY | SET-PARAMS, GET-PARAMS, SPEAK |
| Fetch-timeout | MANDATORY | SET-PARAMS, GET-PARAMS, SPEAK |
| Failed-uri | MANDATORY | Any |
| Failed-uri-cause | MANDATORY | Any |
| Speak-restart | MANDATORY | CONTROL |
| Speak-length | MANDATORY | SPEAK, CONTROL |

9.3 语音识别消息头、方法和事件格式

9.3.1 语音识别消息头

语音识别消息头的格式及其参数如下。

recognizer-header = Confidence-threshold ; 语音识别置信度阈值
/ sensitivity-level ; 语音识别引擎对输入敏感度水平
/ speed-vs-accuracy ; 语音识别引擎计算速度设置

/ n-best-list-length ; 语音识别结果多候选列表长度
/ no-input-timeout ; 无输入检测超时
/ recognition-timeout ; 识别时间过长
/ waveForm-url ; 输入语音文件路径
/ completion-cause ; 结束原因
/ recognizer-context-block ; 语音识别上下文环境信息
/ recognizer-start-timers ; 语音识别启动表记
/ vendor-specific ; 自定义参数
/ speech-complete-timeout ; 获得有效识别结果时, 静音等待时长
/ speech-incomplete-timeout ; 获得无效识别结果是, 静音等待时长
/ dtmf-interdigit-timeout ; DTMF 按键间距
/ dtmf-term-timeout ; DTMF 等待结束符时长
/ dtmf-term-char ; DTMF 结束符
/ fetch-timeout ; 外部资源获取超时
/ failed-uri ; 导致资源获取失败的目标路径
/ failed-uri-cause ; 导致资源获取失败的原因
/ save-waveForm ; 保存语音文件到指定路径
/ speech-language ; 语音识别语言设置
/ new-audio-channel ; 启用新频道识别, 避免上下文环境信息干扰

9.3.2 语音识别方法

语音识别方法的格式及其参数如下。

recognizer-method = SET-PARAMS ; 参数设置
/ GET-PARAMS ; 参数获取
/ DEFINE-GRAMMAR ; 语音识别文法定义
/ RECOGNIZE ; 开始语音识别
/ GET-RESULT ; 获取识别结果
/ RECOGNITION-START-TIMERS ; 延时识别启动
/ STOP ; 停止语音识别

9.3.3 语音识别事件

语音识别事件的格式及其参数如下。

recognizer-event = START-OF-SPEECH ; 静音检测结束, 开始语音识别
/ RECOGNITION-COMPLETE ; 语音识别完成

9.3.4 语音识别消息头与方法、事件的关系

语音识别消息头与方法、事件的关系见表 48。

表 48 语音识别消息头与方法、事件的关系

| 信息头 | 是否支持 | 方法 / 事件 / 响应 |
|----------------------|-----------|-----------------------------------|
| Confidence-threshold | MANDATORY | SET-PARAMS, RECOGNIZE, GET-RESULT |
| Sensitivity-level | Optional | SET-PARAMS, GET-PARAMS, RECOGNIZE |

表 48 (续)

| 信息头 | 是否支持 | 方法 / 事件 / 响应 |
|---------------------------|-----------|--|
| Speed-vs-accuracy | Optional | SET-PARAMS, GET-PARAMS,RECOGNIZE |
| N-best-list-length | Optional | SET-PARAMS, GET-PARAMS,RECOGNIZE, GET-RESULT |
| No-input-timeout | MANDATORY | SET-PARAMS, GET-PARAMS,RECOGNIZE |
| Recognition-timeout | MANDATORY | SET-PARAMS, GET-PARAMS,RECOGNIZE |
| WaveForm-url | MANDATORY | RECOGNITION-COMPLETE |
| Completion-cause | MANDATORY | DEFINE-GRAMMAR, RECOGNIZE,RECOGNITON-COMPLETE |
| Recognizer-context-block | Optional | SET-PARAMS, GET-PARAMS |
| Recognizer-start-timers | MANDATORY | RECOGNIZE |
| Vendor-specific | MANDATORY | SET-PARAMS, GET-PARAMS |
| Speech-complete-timeout | MANDATORY | SET-PARAMS, GET-PARAMS,RECOGNIZE |
| Speech-incomplete-timeout | MANDATORY | SET-PARAMS, GET-PARAMS,RECOGNIZE |
| Dtmf-interdigit-timeout | MANDATORY | SET-PARAMS, GET-PARAMS,RECOGNIZE |
| Dtmf-term-timeout | MANDATORY | SET-PARAMS, GET-PARAMS,RECOGNIZE |
| Dtmf-term-char | MANDATORY | SET-PARAMS, GET-PARAMS,RECOGNIZE |
| Fetch-timeout | MANDATORY | SET-PARAMS, GET-PARAMS,RECOGNIZE, DEFINE-GRAMMAR |
| Failed-uri | MANDATORY | DEFINE-GRAMMAR response,RECOGNITION-COMPLETE |
| Failed-uri-cause | MANDATORY | DEFINE-GRAMMAR response,RECOGNITION-COMPLETE |
| Save-waveForm | MANDATORY | SET-PARAMS, GET-PARAMS,RECOGNIZE |
| New-audio-channel | MANDATORY | RECOGNIZE |
| Speech-languag | MANDATORY | SET-PARAMS, GET-PARAMS,RECOGNIZE, DEFINE-GRAMMAR |

10 语音服务系统安全性要求

10.1 语音认证服务功能

为了防范对用户信息的非法访问，可采用基于语音的说话人身份认证技术，这是语音上网应用领域一项特殊的安全保障手段，用于检测登录用户真伪，拒绝非法信息访问，保护用户信息安全。

在涉及个人隐私信息的各类语音访问中，说话人身份的语音认证可以有效地防范他人通过身份伪装的方式而进行的系统攻击，由于声音是一个人独一无二的个体特征，因此利用声纹特征对访问者进行身份认证，使得冒充者无法通过声音特征检测这一关，从而使系统拒绝冒充者的访问请求，保护用户信息不被泄漏。

语音认证服务功能的提供，需要事先采集用户的声音特征信息，存储在语音服务系统中。在用户登录时，服务系统将现场采集的声音与预先存储的声音比对分析，判别登录者是否是合法用户，做出接受或拒绝访问的处理。

语音认证功能由语音服务器支持，用户的语音信息通过语音网关/语音浏览器上传到语音服务器进行处理。

10.2 安全模块功能要求

10.2.1 安全模块类型

语音服务器上，语音认证功能应包含两个功能模块，即语音模型建立模块和语音身份认证模块。

10.2.2 语音模型建立模块的功能要求

语音模型建立模块用于抽取说话人特征，为新用户建立声音档案，一般需要用户预先提供一些语音片断用于建模。语音认证技术是一个与语言无关、文本无关的检测技术，因此它允许用户随便留下任何内容的声音片断，用于建立声音档案。由于模型质量关系着说话人确认技术的精度，声音片断长度越长，意味着其中包含的声学信息越充分。但考虑到用户体验等方面，语音上网系统一般要求用户留下一段40~60s 时长的声音来满足其精度要求。

声音采集方式可根据语音服务器提供的功能和语音应用程序而制定。在实时的声音录制中，语音应用程序可以根据自身特点来引导用户的录制，可以采用以下方式：

- a) 请求用户在提示音响起后，自由发挥录制一定长度的语音；
- b) 可以先播放一段提示，要求用户跟着重复提示内容；
- c) 可以通过与客户问答交流，自然的收集声音信息，以达到较好的用户体验；
- b) 和 c) 两种方式要求语音服务器可以支持语音累积功能。

在语音服务器可以支持实时的模型训练模式的情况下，可以利用 VoiceXML <Record>元素进行实时的声音录制，再调用语音服务器提供的接口方法，把采集到的声音传递给语音服务器，从而驱动语音服务器完成模型训练功能。在 VoiceXML 中，通过<Subdialog>、<Object>等元素可以完成把<Record>元素的结果做为参数，传递给外部程序（即对语音服务器接口的调用）。

语音服务器建立语音模型时，也可以采用其他方式，可以开发 Web 页面，要求用户上传自己的声音文件到固定位置，再运行后台程序集中进行模型训练。这是一种非实时的离线处理方式，可以减轻业务繁忙时段服务器的压力，但用户使用很不方便。

10.2.3 语音身份认证模块的功能要求

语音身份认证模块运作方式与语音模型建立模块类似，也需要语音系统通过 VoiceXML 程序或者其他手段采集访问者的声音片断，并调用语音服务器的对应接口。

身份认证模块采用当前访问者的声音与其先前存储的纪录比对，提供认证结果。参考目前国际上说话人确认技术的发展水平，认证过程一般仅需要 3~5s 的语音片段就可以得出“通过”或者“拒绝”的认证结果。这类语音片断可以是任意内容，与模型建立模块中使用的语音可以没有任何内容关联性。

在使用认证功能中，VoiceXML <Record>、<Subdialog>、<Object>等元素仍可以提供辅助，用于声音采集，调用语音服务器接口等操作。

针对说话人身份认证功能，各语音服务器厂商会提供自己的接口方式，但都会包含以上提到的两类功能调用接口。考虑到语音上网的应用场景，前述 VoiceXML 语言特点，语音服务器厂商应提供可实时完成说话人身份认证的产品，提供易于与 VoiceXML 程序结合的接口方式，减少用户二次开发的代价。

附 录 A
(资料性附录)
Voice XML 应用示例

A.1 Form 元素应用示例

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
    http://www.w3.org/TR/voicexml20/vxml.xsd">
  <form id="weather_info">
    <block>Welcome to the weather information service.</block>
    <field name="state">
      <prompt>What state?</prompt>
      <grammar src="state.grxml" type="application/srgs+xml"/>
      <catch event="help">
        Please speak the state for which you want the weather.
      </catch>
    </field>
    <field name="city">
      <prompt>What city?</prompt>
      <grammar src="city.grxml" type="application/srgs+xml"/>
      <catch event="help">
        Please speak the city for which you want the weather.
      </catch>
    </field>
    <block>
      <submit next="/servlet/weather" namelist="city state"/>
    </block>
  </form>
</vxml>
```

该对话执行的顺序为：

C (computer): Welcome to the weather information service. What state?

H (human): Help.

C: Please speak the state for which you want the weather.

H: Georgia.

C: What city?

H: Tblisi.

C: I did not understand what you said. What city?

H: Macon.

C: The conditions in Macon Georgia are sunny and clear at 11 AM ...

A.2 Field 元素应用示例

<Field>中可以采用<Grammar>或者<Option>对用户可输入的可选项进行控制。

1) 使用显式 external grammar:

```
<field name="flavor">
  <prompt>What is your favorite ice cream?</prompt>
  <grammar src="../../grammars/ice_cream.grxml" type="application/srgs+xml"/>
</field>
```

2) 使用内嵌 inline grammar:

```
<field name="flavor">
  <prompt>What is your favorite flavor?</prompt>
  <help>Say one of vanilla, chocolate, or strawberry.</help>
  <grammar mode="voice" type="application/srgs">
    #ABNF 1.0;
    $options = vanilla | chocolate | strawberry
  </grammar>
</field>
```

3) 使用内置式 built-in grammar:

```
<grammar src="builtin:grammar/boolean"/>
<grammar src="builtin:dtmf/boolean"/>
```

或简写为:

```
<field type="sample">
  <prompt>Prompt for builtin grammar</prompt>
</field>
```

4) 使用<Option>列表

当要为<Field>指定一组简单的可选项时, 用<Option>列表比用一个语法方便。<Option>列表由包含于<Field>元素中的一组<Option>元素组成

以下例子中的<Field>元素给用户提供了 3 个选项, 把被选定的 application.lastresult\$元素的 value 属性值赋给变量 maincourse:

```
<field name="maincourse">
  <prompt>
    Please select an entree. Today, we are featuring <enumerate/>
  </prompt>
  <option dtmf="1" value="fish"> swordfish </option>
```

```

<option dtmf="2" value="beef"> roast beef </option>
<option dtmf="3" value="chicken"> frog legs </option>
<filled>
  <submit next="/cgi-bin/maincourse.cgi" method="post" namelist="maincourse"/>
</filled>
</field>

```

该例子可能的流程:

C: Please select an entree. Today, we're featuring swordfish; roast beef; frog legs.

H: frog legs.

C: (assigns "chicken" to "maincourse", then submits "maincourse=chicken" to /maincourse.cgi)

A.3 Record 元素应用示例

<Record>应用示例如下:

```

<record name="msg" beep="true" maxtime="10s" finalsilence="4000m
  dtmfterm="true" type="audio/x-wav">
  <prompt timeout="5s">
    Record a message after the beep.
  </prompt>
  <noinput>
    I didn't hear anything, please try again.
  </noinput>
</record>

```

A.4 Object 元素应用示例

<Object>可以对 Java 程序包进行调用, 例如:

```

<object name="debit" classid="method://credit_card/gather_and_debit"
  data="http://www.recordings.example.com/prompts/credit/jesse.jar">
  <param name="amount" expr="document.amt"/>
  <param name="vendor" expr="vendor_num"/>
</object>

```

A.5 Subdialog 元素应用示例

在下面的例子中, 个人的生日用于验证他们的驾驶执照。<Subdialog>元素的 src 指定了同一个文档中的另一个 Dialog 作为 Subdialog, <Param>元素用于传递生日给 Subdialog:

```

<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml

```

```

    http://www.w3.org/TR/voicexml20/vxml.xsd">
<!-- form dialog that calls a subdialog -->
<form>
  <subdialog name="result" src="#getdriverslicense">
    <param name="birthday" expr="2000-02-10"/>.
    <filled>
      <submit next="http://myservice.example.com/cgi-bin/process"/>
    </filled>
  </subdialog>
</form>

<!-- subdialog to get drivers license -->
<form id="getdriverslicense">
  <var name="birthday"/>
  <field name="drivelicense">
    <grammar src="http://grammarlib/drivegrammar.grxml"
      type="application/srgs+xml"/>
    <prompt> Please say your drivers license number. </prompt>
    <filled>
      <if cond="validdrivelicense(drivelicense,birthday)">
        <var name="status" expr="true"/>
      <else/>
        <var name="status" expr="false"/>
      </if>
      <return namelist="drivelicense status"/>
    </filled>
  </field>
</form>
</vxml>

```

A.6 Filled 元素应用示例

<Filled>元素指定了当一个或多个 Form 输入元素被填充后要执行的操作。它可以出现在两个地方：作为<Form>元素的子元素，或者作为输入元素的子元素。例如：

```

<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
    http://www.w3.org/TR/voicexml20/vxml.xsd">

```



```

<form id="get_city">
  <field name="city">
    <grammar type="application/srgs+xml"
      src="http://www.ship-it.example.com/grammars/served_cities.grxml"/>
    <prompt>What is the city?</prompt>
    <filled>
      <if cond="city == 'Novosibirsk'">
        <prompt>Note, Novosibirsk service ends next year.</prompt>
      </if>
    </filled>
  </field>
</form>
</vxml>

```

A.7 Block 元素应用示例

以下<Block>示例表示一段指定的语音输出。

```

<block>
  Welcome to Flamingo, your source for lawn ornaments.
</block>

```

A.8 Initial 元素应用示例

Initial 应用示例如下：

```

<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
    http://www.w3.org/TR/voicexml20/vxml.xsd">
  <form id="weather_info">
    <grammar src="cityandstate.grxml" type="application/srgs+xml"/>
    <!-- Caller can't barge in on today's advertisement. -->
    <block>
      <prompt bargein="false">
        Welcome to the weather information service.
        <audio src="http://www.online-ads.example.com/wis.wav"/>
      </prompt>
    </block>

    <initial name="start">

```

```

<prompt>
    For what city and state would you like the weather?
</prompt>
<help>
    Please say the name of the city and state for which you would like a weather report.
</help>
<!-- If user is silent, reprompt once, then try directed prompts. -->
<noinput count="1"><reprompt/></noinput>
<noinput count="2"><reprompt/><assign name="start" expr="true"/></noinput>
</initial>

<field name="state">
    <prompt>What state?</prompt>
    <help>
        Please speak the state for which you want the weather.
    </help>
</field>

<field name="city">
    <prompt>Please say the city in <value expr="state"/>for which you want the weather.</prompt>
    <help>Please speak the city for which you want the weather.</help>
    <filled>
        <!-- Most of our customers are in LA. -->
        <if cond="city == 'Los Angeles' && state == undefined">
            <assign name="state" expr="'California'"/>
        </if>
    </filled>
</field>

<field name="go_ahead" modal="true">
    <grammar type="application/srgs+xml" src="/grammars/boolean.grxml"/>
    <prompt>Do you want to hear the weather for <value expr="city"/>, <value expr="state"/>?</prompt>
    <filled>
        <if cond="go_ahead">
            <prompt bargein="false">
                <audio src="http://www.online-ads.example.com/wis2.wav"/>
            </prompt>

```

```

        <submit next="/servlet/weather" namelist="city state"/>
    </if>
    <clear namelist="start city state go_ahead"/>
</filled>
</field>
</form>
</vxml>

```

这个例子可以用于如下的交互流程：

C: Welcome to the weather information service. Buy Joe's Spicy Shrimp Sauce.

C: For what city and state would you like the weather?

H: Uh, California.

C: Please say the city in California for which you want the weather.

H: San Francisco, please.

C: Do you want to hear the weather for San Francisco, California?

H: No

C: For what city and state would you like the weather?

H: Los Angeles.

C: Do you want to hear the weather for Los Angeles, California?

H: Yes

C: Don't forget, buy Joe's Spicy Shrimp Sauce tonight!

C: Mostly sunny today with highs in the 80s. Lows tonight from the low 60s ...

<Initial>元素的属性：

- **name** 该 Form Item 变量的名称，用于检测该<Initial>元素是否为可选定的。默认为一个不可访问的内部变量。
- **expr** 该 Form Item 变量的初始值。默认为 ECMAScript 的 Undefined。如果赋一个初始值给它，该 Form ITEM 将不会被访问，除非该 Form Item 变量被清零。
- **cond** ECMAScript 表达式。只有当其结果值为 Boolean True，该代码块才被执行，否则不被执行。缺省时为 True。

A.9 Menu 元素应用示例

Menu 应用示例如下：

```

<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.w3.org/2001/vxml
        http://www.w3.org/TR/voicexml20/vxml.xsd">
    <menu>
        <prompt>

```

```

    Welcome home. Say one of: <enumerate/>
  </prompt>
  <choice next="http://www.sports.example.com/vxml/start.vxml">
    Sports
  </choice>
  <choice next="http://www.weather.example.com/intro.vxml">
    Weather
  </choice>
  <choice next="http://www.stargazer.example.com/voice/astronews.vxml">
    Stargazer astrophysics news
  </choice>
  <noinput>Please say one of <enumerate/></noinput>
</menu>
</vxml>

```

这个示例中给用户提供了 3 种输入选项，其可能的交互流程为：

C: Welcome home. Say one of: sports; weather; Stargazer astrophysics news.

H: Astrology.

C: I did not understand what you said. (a platform-specific default message.)

C: Welcome home. Say one of: sports; weather; Stargazer astrophysics news.

H: sports.

C: (proceeds to <http://www.sports.example.com/vxml/start.vxml>)

A.10 Enumerate 元素应用示例

Enumerate 应用示例如下：

```

<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
    http://www.w3.org/TR/voicexml20/vxml.xsd">
  <menu dtmf="true">
    <prompt>
      Welcome home.
      <enumerate>
        For <value expr="_prompt"/>, press <value expr="_dtmf"/>.
      </enumerate>
    </prompt>
    <choice next="http://www.sports.example.com/vxml/start.vxml">
      sports </choice>

```

```

    <choice next="http://www.weather.example.com/intro.vxml">
      weather </choice>
    <choice next="http://www.stargazer.example.com/voice/astronews.vxml">
      Stargazer astrophysics news </choice>
  </menu>
</vxml>

```

该 Menu 的提示语为：

C: Welcome home. For sports, press 1. For weather, press 2. For Stargazer astrophysics news, press 3.

A.11 Link 元素应用示例

在以下示例中，当用户说“Books”或按“2”时，<Link>元素被激活：

```

<link next="http://www.voicexml.org/books/main.vxml">
  <grammar mode="voice" version="1.0" root="root">
    <rule id="root" scope="public">
      <one-of>
        <item>books</item>
        <item>VoiceXML books</item>
      </one-of>
    </rule>
  </grammar>
  <grammar mode="dtmf" version="1.0" root="r2">
    <rule id="r2" scope="public"> 2 </rule>
  </grammar>
</link>

```

A.12 Grammer 元素应用示例

<Grammar>元素也可以用来提供 DTMF 语法，下面是一个简单的 XML 格式的 DTMF 联机语法，它只接受按键“123”或“#”。

```

<grammar mode="dtmf" version="1.0" root="root">
  <rule id="root" scope="public">
    <one-of>
      <item> 1 2 3 </item>
      <item> # </item>
    </one-of>
  </rule>
</grammar>

```

A.13 Prompt 元素应用示例

1) Prompt 的基本应用

```
<prompt>
  Welcome to the Bird Seed Emporium.
  <audio src="rtsp://www.birdsounds.example.com/thrush.wav"/>
  We have 250 kilogram drums of thistle seed for
  <say-as interpret-as="currency">$299.95</say-as>
  plus shipping and handling this month.
  <audio src="http://www.birdsounds.example.com/mourningdove.wav"/>
</prompt>
```

2) Prompt 的 Barge-in

如果执行平台支持 Bargein, 则开发者就能够指定用户是否可以使用 DTMF 或语音打断正在播放的提示语, 这样可以加快交互的速度, 例如:

```
<prompt bargein="false"><audio src="legalese.wav"/></prompt>
```

3) Prompt 的分级提示

分级提示即每次播放的提示语都可以不一样。当用户对服务更熟悉时, 提示语可变得更简短; 而用户需要更多的帮助时, 提示语可变得更详细; 或者提示语不停地改变只是为了使交互更有趣。

例如, 下面例子中的 Form 中有一个 Form 级的<Prompt>元素和几个 Filled 级的<Prompt>元素:

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
    http://www.w3.org/TR/voicexml20/vxml.xsd">
  <form id="tapered">
    <block>
      <prompt bargein="false">
        Welcome to the ice cream survey.
      </prompt>
    </block>
    <field name="flavor">
      <grammar mode="voice" version="1.0" root="root">
        <rule id="root" scope="public">
          <one-of>
            <item>vanilla </item>
            <item>chocolate </item>
            <item>strawberry </item>
          </one-of>
```

```

        </rule>
    </grammar>
    <prompt count="1">What is your favorite flavor?</prompt>
    <prompt count="3">Say chocolate, vanilla, or strawberry.</prompt>
    <help>Sorry, no help is available.</help>
</field>
</form>
</vxml>

```

其可能对应的流程为：

C: Welcome to the ice cream survey.
 C: What is your favorite flavor? (the "flavor" field's prompt counter is 1)
 H: Pecan praline.
 C: I do not understand.
 C: What is your favorite flavor? (the prompt counter is now 2)
 H: Pecan praline.
 C: I do not understand.
 C: Say chocolate, vanilla, or strawberry. (prompt counter is 3)
 H: What if I hate those?
 C: I do not understand.
 C: Say chocolate, vanilla, or strawberry. (prompt counter is 4)
 H: ...

A.14 Throw 元素应用示例

<Throw>元素用于抛出一个事件，可以抛出预定义或自定义的事件，例如：

```

<throw event="nomatch"/>
<throw event="connection.disconnect.hangup"/>
<throw event="com.att.portal.machine"/

```

A.15 Catch 元素应用示例

<Catch>元素将文档、对话、Form Item 与事件捕获联系起来，它包含了可执行的内容，见如下例子：

```

<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
    http://www.w3.org/TR/voicexml20/vxml.xsd">
  <form id="launch_missiles">
    <field name="user_id" type="digits">
      <prompt>What is your username</prompt>

```

```

</field>
<field name="password">
  <prompt>What is the code word?</prompt>
  <grammar version="1.0" root="root">
    <rule id="root" scope="public">rutabaga</rule>
  </grammar>
  <help>It is the name of an obscure vegetable.</help>
  <catch event="nomatch noinput" count="3">
    <prompt>Security violation!</prompt>
    <submit next="http://www.example.com/apprehend_felon.vxml"
      namelist="user_id"/>
  </catch>
</field>
</form>
</vxml>

```

<Catch>元素的匿名变量包括一个专有的变量 “_event”，它保存了被捕获的事件名。例如，下面的<Catch>元素可以处理两种事件：

```

<catch event="event.foo event.bar">
  <if cond="_event=='event.foo'">
    <!-- Play this for event.foo events -->
    <audio src="foo.wav"/>
  <else/>
    <!-- Play this for event.bar events -->
    <audio src="bar.wav"/>
  </if>
  <!-- Continue with common handling for either event -->
</catch>

```

上面的 “_event” 变量的作用是根据被捕获的事件检查该播放哪个音频。如果 event.foo 事件被捕获，就播放 foo.wav；如果 event.bar 事件被捕获，就播放 bar.wav。<Catch>元素包含的剩下的可执行内容对两类事件的处理都是一样的。

<Catch>元素的匿名变量也包括专有变量 “_message”，它保存了来自相应的<Throw>元素的 Message 字符串的值，或者平台为预定义事件产生的依赖于平台的值。如果被抛出的事件没有指定 Message，则该 “_message” 的值为 ECMAScript 的 Undefined。

如果<Catch>元素包含了一个<Throw>，且<Catch>要捕获的事件和<Throw>元素抛出的事件一样的话，会形成一个无限循环：

```

<catch event="help">
  <throw event="help"/>

```



```
</catch>
```

平台要能够检测到这种情况，并抛出一个语义错误。

A.16 Catch 元素的选择示例

如果抛出的事件名匹配了<Catch>元素中的事件名，则这是一个精确匹配，或是前缀匹配，或者该<Catch>元素没有指定 Event 属性，如果<Catch>元素 Event 属性值中的事件是所抛出的事件的前缀，则为前缀匹配。点号是前缀的分隔符，点号后面的字符都去掉。空的字符串可以匹配任何事件。例如：

```
<catch event="connection.disconnect">
  <prompt>Caught a connection dot disconnect event</prompt>
</catch>
```

在该例子中，事件 connection.disconnect 前缀匹配事件 connection.disconnect.event1。

在<Catch>元素的选定算法中，文档顺序靠前的<Catch>元素的优先权比文档顺序靠后的高。该算法并没有规定，指定的事件越明确的<Catch>元素的优先权越高，因此在一般情况下，我们建议，指定的事件越明确的<Catch>元素，其文档顺序应越靠前；而越不明确的<Catch>元素，其文档顺序应越靠后。例如，建议捕捉事件“error.foo”和“error”的文档顺序如下：

```
<catch event="error.foo">
  <prompt>Caught an error dot foo event</prompt>
</catch>
<catch event="error">
  <prompt>Caught an error event</prompt>
</catch>
```

如果上述的<Catch>元素的文档顺序相反，则捕捉事件“error.foo”的<Catch>元素永远都不会被执行。

一个元素如果需要的话，将从它的每个上层元素继承(“as if by copy”)<Catch>元素。例如，如果<Field>元素从文档继承<Catch>元素：

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
    http://www.w3.org/TR/voicexml20/vxml.xsd">
  <catch event="event.foo">
    <audio src="beep.wav"/>
  </catch>

  <form>
    <field name="color">
      <prompt>Please say a primary color</prompt>
      <grammar type="application/srgs">red | yellow | blue</grammar>
      <nomatch>
```

```

        <throw event="event.foo"/>
    </nomatch>
</field>
</form>
</vxml>

```

则该<Catch>元素被隐式的拷贝到该<Field>元素中，等同于如下写法：

```

<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.w3.org/2001/vxml
        http://www.w3.org/TR/voicexml20/vxml.xsd">
    <form>
        <field>
            <prompt>Please say a primary color</prompt>
            <grammar type="application/srgs">red | yellow | blue</grammar>
            <nomatch>
                <throw event="event.foo"/>
            </nomatch>
            <catch event="event.foo">
                <audio src="beep.wav"/>
            </catch>
        </field>
    </form>
</vxml>

```

A.17 快捷标记示例

<Error>、<Help>、<Noinput>和<Nomatch>元素都是<Catch>元素特定形式的快捷标记，例如：

```

<error> An error has occurred -- please call again later. </error>
<help>No help is available.</help>
<noinput>I didn't hear anything, please try again.</noinput>
<nomatch>I heard something, but it wasn't a known city.</nomatch>

```

A.18 Var 元素应用示例

<Var>可以出现在任何可执行的内容中，也可以作为<Form>元素或<Vxml>元素的子元素出现。例如：

```

<var name="phone" expr="6305551212"/>
<var name="y" expr="document.z+1"/>

```

A.19 Assign 元素应用示例

<Assign>元素用来给变量赋值，例如：

```
<assign name="flavor" expr="chocolate"/>
<assign name="document.mycost" expr="document.mycost+14"/>
```

A.20 Clear 元素应用示例

<Clear>元素的作用是重置（清零）一个或多个变量，包括 Form Item，例如：

```
<clear namelist="city state zip"/>
```

A.21 IF, ELSEIF, ELSE 应用示例

<If>元素用于判断条件逻辑，它可以和<Elseif>、<Else>搭配使用，例如：

```
<if cond="total > 1000">
    <prompt>This is way too much to spend.</prompt>
</if>
```

```
<if cond="amount < 29.95">
    <assign name="x" expr="amount"/>
<else/>
    <assign name="x" expr="29.95"/>
</if>
```

```
<if cond="flavor == 'vanilla'">
    <assign name="flavor_code" expr="v"/>
<elseif cond="flavor == 'chocolate'">
    <assign name="flavor_code" expr="h"/>
<elseif cond="flavor == 'strawberry'">
    <assign name="flavor_code" expr="b"/>
<else/>
    <assign name="flavor_code" expr="?" />
</if>
```

A.22 Prompt

提示语可以采用<Prompt>元素的一般形式出现在可执行内容中，除非不使用<Prompt>元素的 Count 属性。特别的，它的 Cond 属性也可用在可执行内容中。提示语可以被封装在<Prompt>和</Prompt>中，或使用 PCDATA 表示。只要是<Prompt>元素允许出现的地方，使用 PCDATA xyz 等价于使用<Prompt>xyz</Prompt>，例如：

```
<nomatch count="1">
    To open the pod bay door, say your code phrase clearly.
</nomatch>
```

A.23 Reprompt 元素应用示例

<Reprompt>用于重新播放提示语，只用于<Catch>元素内，例如：

```
<field name="want_ice_cream">
  <grammar type="application/srgs+xml" src="/grammars/boolean.grxml"/>
  <prompt>Do you want ice cream for dessert?</prompt>
  <prompt count="2">
    If you want ice cream, say yes.
    If you do not want ice cream, say no.
  </prompt>
  <noinput>
    I could not hear you.
    <!-- Cause the next prompt to be selected and played. -->
  <reprompt/>
</noinput>
</field>
```

可能的一个对话流程：

C: Do you want ice cream for dessert?

H: (silence)

C: I could not hear you.

C: If you want ice cream, say yes. If you don't want ice cream, say no.

H: (silence)

C: I could not hear you.

C: If you want ice cream, say yes. If you don't want ice cream, say no.

H: No.

如果该例子中没有<Reprompt>元素，上面的流程变为如下所示：

C: Do you want ice cream for dessert?

H: (silence)

C: I could not hear you.

H: (silence)

C: I could not hear you.

H: No.

A.24 Goto 元素应用示例

要跳转到另一个 Form Item，使用 Next 属性或使用 Expritem 属性，如果该 Form Item 名可以通过 ECMAScript 表达式计算得到的话，如：

```
<goto nextitem="ssn_confirm"/>
<goto expritem="(type==12)? 'ssn_confirm' : 'reject'"/>
```

要跳转到同一个文档的另一个 Dialog，使用 Next 属性，或使用 Expr 属性指定一个 URI：

```
<goto next="#another_dialog"/>
<goto expr="#" + 'another_dialog' />
```

要跳转到另一个文档使用 Next 属性或使用 Expr 属性指定一个 URI：

```
<goto next="http://flight.example.com/reserve_seat"/>
<goto next="/special_lunch#wants_vegan"/>
```

URI 可以是当前文档的绝对或相对的 URI。要指定下一个文档开始的 Dialog，可以使用对应于该 Dialog 的 ID 属性值的 URI 段。如果没有指定 URI 段，则以要跳转到的文档的第一个 Dialog 为开始的 Dialog。

跳转到当前文档的另一个 Dialog 会导致旧的 Dialog 变量丢失，即使是一个 Dialog 跳转到它自己也会丢失。同样的，使用相对或绝对的 URI 跳转到另一个文档，也会丢失文档变量，即使是一个文档跳转到它自己也会丢失。然而，当跳转到一个带有段标识符的空的 URI 引用时，文档变量不会丢失。例如下面两个语句在 URI 为 <http://someco.example.com/index.vxml> 的文档中有不同的行为：

```
<goto next="#foo"/>
<goto next="http://someco.example.com/index.vxml#foo"/>
```

原因是，根据 IETF RFC2396 中，段标识符（“#”后面的部分）不是 URI 的一部分，到一个空的 URI 引用加上段标识符的跳转永远都不会获取到一个新的文档，因此，第一条语句中的“#foo”是一个带有段标识符的空的 URI 引用，它的文档变量不会丢失。在第二条语句中，“#foo”是一个绝对 URI 的一部分，该跳转会导致文档变量的丢失。如果要在多文档中保存数据，应把数据保存在应用根文档中。

A.25 Submit 元素应用示例

<Submit>元素用于提交信息给 Web 服务器，然后跳转到 Web 服务器响应后返回的文档。和<Goto>元素不一样的是，<Submit>元素可以通过 HTTP GET 或 POST 请求提交一个变量列表到文档服务器。例如可以提交一组 Form Item 变量到服务器。

```
<submit next="log_request" method="post"
  namelist="name rank serial_number"
  fetchtimeout="100s" fetchaudio="audio/brhams2.wav"/>
```

<Submit>元素的 Next 或 Expr 属性指定了要跳转到的 Dialog。该 URI 每次都要被获取，即使它只包含了一个段，因此，下面的两个元素有完全不同的效果：

```
<goto next="#get_pin"/>
<submit next="#get_pin"/>
```

A.26 Return 元素应用示例

下面的例子展示了，当 Subdialog 不能获得一个可识别的结果时，一个 Nomatch 事件通过 Subdialog 在调用它的 Dialog 中抛出。

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```

xsi:schemaLocation="http://www.w3.org/2001/vxml
http://www.w3.org/TR/voicexml20/vxml.xsd">
<form>
  <subdialog name="result" src="#getssn">
    <nomatch>
      <!-- a no match event that is returned by the subdialog indicates
            that a valid social security number could not be matched. -->
      <goto next="http://myservice.example.com/ssn-problems.vxml"/>
    </nomatch>

    <filled>
      <submit namelist="result.ssn"
        next="http://myservice.example.com/cgi-bin/process"/>
    </filled>
  </subdialog>
</form>
<form id="getssn">
  <field name="ssn">
    <grammar src="http://grammarlib/ssn.grxml" type="application/srgs+xml"/>
    <prompt> Please say social security number.</prompt>
    <nomatch count="3">
      <return event="nomatch"/>
    </nomatch>
    <filled>
      <return namelist="ssn"/>
    </filled>
  </field>
</form>
</vxml>

```

Subdialog 中的 Nomatch 事件处理在第 3 次匹配失败时被触发, 然后从 Subdialog 中返回, 并在调用它的 Dialog 中抛出该事件。在这种情况下, 调用的 Dialog 将执行它自己的 Nomatch 事件处理, 通过<Goto>元素跳转到另一个文档, 而不是执行<Filled>操作。在一般的情况下, 如果得到一个识别结果, 就会执行该<Subdialog>元素中的<Filled>操作, 该识别结果可通过 result.ssn 访问。

A.27 Script 元素应用示例

<Script>元素允许在 VoiceXML 脚本中使用一段客户端的脚本语言代码, 它和 HTML 的<Script>元素类似。例如下面的例子中<Script>元素用于计算一个数的阶乘:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3.org/2001/vxml
      http://www.w3.org/TR/voicexml20/vxml.xsd">
  <script> <![CDATA[
    function factorial(n)
    {
      return (n <= 1)? 1 : n * factorial(n-1);
    }
  ]]> </script>

  <form id="form">
    <field name="fact">
      <nomatch>
        <grammar type="application/srgs+xml" src="/grammars/number.grxml"/>
        <prompt>
          Tell me a number and I'll tell you its factorial.
        </promptd>
        <filled>
          <prompt>
            <value expr="fact"/> factorial is
            <value expr="factorial(fact)"/>
          </prompt>
        </filled>
      </field>
    </form>
  </vxml>

```

A.28 Log 元素应用示例

Log 元素应用示例如下：

```
<log>The card number was <value expr="card_num"/></log>
```

A.29 Meta 元素应用示例

<Meta>元素指定了元信息，有两种用法：

第 1 种是指定整个文档的元数据 Property，它由 Name 和 Content 这对属性表示。例如，指定 VoiXML 文档的维护人员：

```
<meta name="maintainer" content="jpdoe@anycompany.example.com"/>
```

第 2 种是指定了 HTTP 响应头，它由 HTTP-Equiv 和 Content 这对属性表示。在下面的例子中，第 1

个<Meta>元素设置了一个过期日期，以防止该文档缓存；第 2 个<Meta>元素设置了日期头。

```
<meta http-equiv="Expires" content="0"/>
<meta http-equiv="Date" content="Thu, 12 Dec 2000 23:27:21 GMT"/>
```

A.30 Metadata 元素应用示例

下面的例子展示了在 VoiceXML 文档中怎样使用<Metadata>元素：

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
    http://www.w3.org/TR/voicexml20/vxml.xsd">
  <metadata>
    <rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#"
      xmlns:dc="http://purl.org/metadata/dublin_core#">

      <!-- Metadata about the VoiceXML document -->
      <rdf:Description about="http://www.example.com/meta.vxml"
        dc:Title="Directory Enquiry Service"
        dc:Description="Directory Enquiry Service for London in VoiceXML"
        dc:Publisher="W3C"
        dc:Language="en"
        dc:Date="2002-02-12"
        dc:Rights="Copyright 2002 John Smith"
        dc:Format="application/voicexml+xml">
        <dc:Creator>
          <rdf:Seq ID="CreatorsAlphabeticalBySurname">
            <rdf:li>Jackie Crystal</rdf:li>
            <rdf:li>William Lee</rdf:li>
          </rdf:Seq>
        </dc:Creator>
      </rdf:Description>
    </rdf:RDF>
  </metadata>
  <form>
    <block>
      <prompt>Hello</prompt>
```



```

    </block>
  </form>
</vxml>

```

A.31 Property 元素应用示例

解释器环境可以任意的提供依赖于平台的 Property。例如下面的例子，在该文档内设置“Multiplication Factor” 属性：

```

<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
    http://www.w3.org/TR/voicexml20/vxml.xsd">
  <form>
    <property name="com.example.multiplication_factor" value="42"/>
    <block>
      <prompt>Welcome</prompt>
    </block>
  </form>
</vxml>

```

下面的例子展示了在各个级别使用 Property。

```

<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
    http://www.w3.org/TR/voicexml20/vxml.xsd">
  <!-- set default characteristics for page -->
  <property name="audiofetchhint" value="safe"/>
  <property name="confidencelevel" value="0.75"/>

  <form>
    <!-- override defaults for this form only -->
    <property name="confidencelevel" value="0.5"/>
    <property name="bargain" value="false"/>
    <grammar src="address_book.grxml" type="application/srgs+xml"/>
    <block>
      <prompt>Welcome to the Voice Address Book</prompt>
    </block>
  </form>
</vxml>

```

```

<initial name="start">
  <!-- override default timeout value -->
  <property name="timeout" value="5s"/>
  <prompt>Who would you like to call?</prompt>
</initial>
<field name="person">
  <prompt>Say the name of the person you would like to call.</prompt>
</field>
<field name="location">
  <prompt>Say the location of the person you would like to call.</prompt>
</field>
<field name="confirm">
  <grammar type="application/srgs+xml" src="/grammars/boolean.grxml"/>
  <!-- Use actual utterances to playback recognized words, rather than returned slot values -->
  <prompt>
    You said to call <value expr="person$.utterance"/>
    at <value expr="location$.utterance"/>.
    Is this correct?
  </prompt>
  <filled>
    <if cond="confirm">
      <submit namelist="person location"
        next="http://www.messagecentral.example.com/voice/make_call" />
    </if>
    <clear>
  </filled>
</field>
</form>
</vxml>

```

A.32 Param 元素应用示例

下面是一个<Object>元素中的<Param>元素的用法的例子。在这个例子中，前两个<Param>元素包含有表达式（隐式的属性值 Valuetype="data"），第 3 个<Param>元素有显式的值，而第 4 个<Param>元素则包含了一个 URI，该 URI 返回一个 Text/Plain 的媒体类型。

```

<object name="debit"
  classid="method://credit_card/gather_and_debit"
  data="http://www.recordings.example.com/prompts/credit/jesse.jar">
  <param name="amount" expr="document.amt"/>

```

```
<param name="vendor" expr="vendor_num"/>
<param name="application_id" value="ADC5678-QWOO"/>
<param name="authentication_server"
  value="http://auth_svr.example.com"
  valuetype="ref"
  type="text/plain"/>
</object>
```

A.33 MRCP 语音合成示例

MRCP 语音合成示例如下:

C->S:SPEAK 543260 MRCP/1.0

Voice-gender:female

Prosody-volume:medium

Speech-language:zh-CN

Content-Type:application/synthesis+ssml

Content-Length:427

```
<?xml version="1.0"?>
```

```
<speak>
```

```
<paragraph>
```

```
<sentence>你有 4 条新信息</sentence>
```

```
<sentence>第一条来自电话 12345678
```

```
<break/>
```

```
  到达时间为<say-as type="time">3:45pm</say-as>.
```

```
</sentence>
```

```
<sentence> 信息内容为
```

```
<prosody rate="-20%"> 国内新闻一览</prosody>
```

```
</sentence>
```

```
</paragraph>
```

```
</speak>
```

S->C:MRCP/1.0 543260 200 IN-PROGRESS

S->C:SPEAK-COMplete 543260 COMplete MRCP/1.0

Completion-Cause:000 normal

A.34 MRCP 语音识别示例

MRCP 语音识别示例如下:

C->S:RECOGNIZE 543257 MRCP/1.0

Confidence-Threshold:90

Content-Type:application/grammar+xml

Content-Id:request1@form-level.store

Content-Length:339

<?xml version="1.0" encoding="UTF-8"?>

<grammar xmlns="http://www.w3.org/2001/06/grammar"

xml:lang="zh-CN"

version="1.0"

root="test">

<rule id="test">

<one-of>

<item>人民币</item>

<item>外币</item>

</one-of>

</rule>

</grammar>

S->C:MRCP/1.0 543257 200 IN-PROGRESS

S->C:START-OF-SPEECH 543257 IN-PROGRESS MRCP/1.0

S->C:RECOGNITION-COMplete 543257 COMPLETE MRCP/1.0

Completion-Cause:000 success

Content-Type:application/x-nlsml

Content-Length:276

content-type: application/x-nlsml; charset=UTF-8

<?xml version="1.0" encoding="UTF-8"?>

<result grammar="session:reco1">

<interpretation confidence="95" grammar="session:reco1">

<instance>

外币

</instance>

<input confidence="95" mode="speech">

外币

</input>

</interpretation>

</result>