Технология VoiceXML и её приложения

Е.Б. Никитин,
аспирант

VoiceXML является мощной и гибкой веб-технологией в области синтеза и распознавания речи. В данной статье описывается история создания, преимущества и способы применения стандартного XML-формата для описания интерактивных диалогов между человеком и компьютером VoiceXML.

Abctract

VoiceXML is a powerful and agile web-technology in the text-to-speech and voice recognition field. The history of its creation, advantages and applications are described in the article. VoiceXML stands for a standard XML format for specifying interactive voice dialogues between a human and a computer.

Введение

VoiceXML — это открытый стандартный язык разметки для голосовых приложений, разработанный организацией VoiceXML Forum [1], которая была создана в 1999 году и объединяет 44 фирмы, среди которых Ericsson, Siemens, France Telecom, Novell, Samsung Electronics, Sun Microsystems, IBM, AT&T, Lucent Technologies и Motorola [2–11]. VoiceXML предназначен для создания и передачи ориентированных на Web персонализированных интерактивных сервисов с речевым ответом и для обеспечения телефонного и речевого доступа к интегрированным базам данных центров обработки вызовов (call center databases), а также к информации на Web-узлах и в интрасетях.

История VoiceXML началась в 1995 году, когда работники AT&T Research Dave Ladd, Chris Ramming, Ken Rehor и Curt Tuckey [12] собирались в неформальной обстановке и обсуждали, как Интернет повлияет на телефонию. Они решили, что нужно создать систему, которая посредством голосового браузера позволяла бы предоставлять контент и услуги пользователям обычных телефонов. Для этого был создан проект AT&T Phone Web project. Потом Lucent откололся от AT&T, и каждая фирма начала разрабатывать свой собственный стандарт. Chris Ramming остался работать в AT&T, Ken Rehor оказался в Lucent, а Dave Ladd и Curt Tuckey перешли в Motorola. К началу 1999 года AT&T и Lucent имели несовместимые разновидности языка разметки звонков (Phone Markup Language — PML), у Motorola был новый стандарт VoxML, другие компании также экспериментировали с идеями интеграции контента и сервисов Интернет в телефонию. Для функционирования голосовых веб-сервисов

требовалась разработка стандартного языка. Создатели Phone Web поспособствовали тому, что AT&T, Lucent и Motorola основали VoiceXML Forum. Вскоре к ним присоединилась IBM. К августу 1999 небольшая команда инженеров Форума выпустила VoiceXML 0.9, в котором были объединены лучшие особенности предыдущих языков, а также добавлены новые идеи — особенно важным было добавление тонального набора (Dual-Tone Multi-Frequency, DTMF) [13]. После этого сообщество расширилось, язык был значительно дополнен новыми функциями — и в марте 2000 года был обнародован VoiceXML 1.0. На следующий день появилось около двадцати различных приложений этого языка.

Вскоре после выхода VoiceXML 1.0 Форум вступил в World Wide Web Consortium (W3C) [14], благодаря чему выпускались промежуточные версии VoiceXML 2.0, которые в марте 2004 дошли до финального этапа создания рекомендаций. Рекомендация VoiceXML 2.1 увидела свет 19 июня 2007 года [15]. Это последняя версия на данный момент.

1. Процесс синтеза речи

Синтез речи производится системой TTS (Text-To-Speech), которая принимает SSML-документы (Speech Synthesis Markup Language) [16], а на выходе даёт голосовой поток.

Процесс переработки XML-файла в голосовое сообщение состоит из шести шагов.

1) Разбор XML: процесс разбора XML используется для извлечения содержания и дерева документа из входящего текстового файла. Структура, тэги и атрибуты, полученные на этом шаге, влияют на все последующие. Словами в SSML не могут быть тэги разметки: например, последовательность «black<break/>berry» обработчик воспримет как два слова: «black» и «berry», — а не как одно слово с паузой в середине. Разбиение на более мелкие элементы может изменить интерпретацию обработчиком.

2) Анализ структуры: от структуры документа зависит, как он будет прочитан. Например, существуют речевые шаблоны, связанные с абзацами и предложениями.

- *С поддержкой разметки:* P- и S-элементы (P — paragraph, абзац; S — sentence, предложение), определённые в SSML, однозначно задают структуру документа.

- *Без разметки:* если эти элементы не использованы в документе или его частях, обработчик сам распознаёт структуру, анализируя текст, часто опираясь на пунктуацию и другие особенности языка.

3) Нормализация текста: в языках существуют специальные конструкции, которые требуют перевода текста из письменной (орфографической) формы в устную. Нормализация текста — это автоматический процесс, который производится обработчиком синтеза. Например, в русском языке текст «$200» произносится как «двести долларов». Аналогично, «1/2» может произноситься как «одна вторая», «половина», «первое февраля» и т.д.

После этого шага текст для перевода в устную речь полностью преобразуется в лексемы. Что стоит за лексемой, зависит от конкретного языка. Так, в английском языке лексемы обычно разделены пробелом и являются словами.

- *С поддержкой разметки:* элемент **say-as** может использоваться для однозначного задания типа конструкций и разрешения неопределённостей.

- *Без разметки:* не размеченный элементами **say-as** текст разбирается автоматически, однако при этом следует ожидать больших трудностей из-за неопределённостей наподобие описанного выше примера с «1/2».

4) Перевод текста в фонемы: после того, как обработчик синтеза определил набор слов, которые должны быть сказаны, необходимо определить произношение каждого слова. Произношение слов может быть описано как последовательность фонем, которые являются звуковыми элементами языка. Например, в английском слово «read» может произноситься как «рэд» или как «рид», в зависимости от временной формы глагола.

- *С поддержкой разметки:* элемент **phoneme** позволяет задавать фонемные последовательности для любого слова, что предоставляет полный контроль над произношением. Также можно использовать элемент **say-as**, чтобы текст считался именем собственным, что позволит обработчику применить специальные правила для определения произношения. Эти элементы являются особенно полезными для акронимов и аббревиатур.

- *Без разметки:* в случае отсутствия элемента phoneme обработчик применяет автоматические функции по определению произношения. Это производится за счёт поиска слов в словаре и применения правил для определения других произношений. Обработчики синтеза речи разработаны так, чтобы производить перевод текста в фонемы, так что большинство слов могут быть обработаны автоматически. Альтернативным способом является изменение текста перед отправкой его обработчику таким образом, чтобы избежать многозначности, например, заменить слово «read» на «reed». При этом надо учитывать, что такой текст лучше нигде не отображать, потому что он грамматически некорректен.

5) Просодический анализ: просодема — это набор особенностей речи, которые включают в себя интонацию, ритм, паузы, скорость речи, ударения и другие. Воспроизведение просодем, близких к естественному языку, является важным, чтобы речь звучала естественно и корректно.

- *С поддержкой разметки:* элементы **emphasis**, **brake** и **prosody** могут быть использованы создателем документа для указания обработчику правильных просодем.

- *Без разметки:* в случае отсутствия этих элементов обработчик сам подберёт необходимые просодемы. Это достигается за счёт анализа структуры документа, синтаксиса и другой информации.

6) Генерация сигнала: фонемы и просодемы используются обработчиком для генерации аудиосигнала. Существует много разных подходов к этому шагу.

- *С поддержкой разметки:* элемент **voice** позволяет создателю документа запрашивать особенный тип голоса (например, голос взрослого мужчины). Элемент audio дает возможность вставлять предварительно записанные аудиофрагменты в выходной поток.

2. Процесс распознавания речи

Распознавание речи осуществляется системой STT (Speech-To-Text). Из-за большого количества возможных произношений и акцентов STT-процесс является зачастую сложной и неоднозначной процедурой. Для упрощения и повышения надёжности работы системы STT применяются грамматики. Грамматика распознавания речи — это набор паттернов, которые подсказывают системе распознавания речи, что сейчас может сказать пользователь. Например, если вы звоните в автоматический справочник, система спросит имя человека, с которым вас нужно соединить. После этого система запустит распознаватель речи, предоставив ему грамматику распознавания речи. Эта грамматика содержит имена людей в справочнике и наиболее частые варианты ответов звонящих. Speech Recognition Grammar Specification (SRGS — спецификация грамматики распознавания речи) [17] является стандартом W3C для описания грамматик распознавания речи.

SRGS определяет два альтернативных, но логически эквивалентных синтаксиса: один основан на XML, другой использует расширенный формат BNF (Backus-Naur Form) [18]. На практике чаще используется XML.

Если бы система распознавания речи возвращала просто строку со словами, сказанными пользователем, голосовому приложению пришлось бы выполнять большую работу по извлечению семантических значений этих слов. Поэтому SRGS-грамматики могут быть оформлены тэгами, в случае использования которых строится семантический результат. SRGS не описывает содержание ярлыков, потому что это сделано в дополнительном стандарте SISR (Semantic Interpretation for Speech Recognition) [19]. VoiceXML тесно связан со стандартами SRGS и SISR.

Пользовательский агент — это обработчик грамматики, который на входе принимает речь и сопоставляет её с грамматикой, а на выходе даёт соответствующий результат распознавания.

Распознаватель речи — это пользовательский агент, использующий следующие входные и выходные параметры:

- *Вход A*: грамматика или множество грамматик. Эти грамматики информируют распознаватель о словах и паттернах слов, которые надо слушать.
- *Вход B*: аудиопоток, который может содержать речь, соответствующую грамматикам.
- *Выход*: описание результатов, которое содержит детальную информацию о распознанном речевом входе. Формат и детали результата являются свободными. Для информативности большинство систем распознавания включают в результаты хотя бы транскрипцию слов, которые были распознаны. Также голосовому браузеру могут быть переданы ошибки и другая информация о работе.

3. Особенности разработки под VoiceXML

Как говорилось ранее, VoiceXML является приложением языка XML и имеет структуру дерева, по которому пользователь может перемещаться с помощью голосовых команд. Первые VoiceXML-платформы строго требовали, чтобы определение типа документа (Document Type Definition — DTD) объявлялось непосредственно перед начальным тэгом <vxml>. Современные VoiceXML-браузеры намного более гибки в этой части, так что теперь нет необходимости

в дополнительном объявлении DTD. Более того, для большей гибкости приложений рекомендуется вообще нигде в коде не указывать DTD. Неотъемлемым компонентом любого VoiceXML-приложения является система распознавания и синтеза речи, установленная на сервере. Доступны подобные продукты от таких известных фирм, как IBM [20], Microsoft [21], Speechtech [22], Voxeo [23]. Классификация тэгов VoiceXML (таблица 1) проста и логична.

Таблица 1

Типы тэгов VoiceXML

Назначение	Тэг	Назначение	Тэг
Определение приложения	<vxml>	Определение грамматик	<grammar>
	<meta>		<rule>
Диалоги	<form>		<ruleref>
	<menu>		<token>
Элементы ввода формы	<field>		<one-of>
	<record>		<item>
	<subdialog>		<tag>
	<transfer>		<example>
Элементы контроля формы	<initial>		<lexicon>
	<block>	Контроль интерпретатора	<property>
Меню	<menu>	Контейнеры	<block>
	<choice>		<filled>
	<grammar>		<if>
	<enumerate>		<catch>
Поля	<field>		<error>
	<enumerate>		<help>
	<grammar>		<noinput>
	<option>		<nomatch>
	<filled>		<object>
	<help>	Объявление и установка переменных	<var>
	<noinput>		<assign>
	<nomatch>		<clear>
Субдиалоги	<subdialog>	Процедурная логика	<if>
	<param>		<else>
	<return>		<elseif>
Контроль перехода в диалогах	<goto>	Логика сценариев	<script>
	<submit>	Создание аудиовыхода	<audio>
	<link>		<prompt>
	<choice>		<reprompt>
Контроль синтеза речи	<break>		<value>
	<voice>		<enumerate>
	<emphasis>	События	<throw>
	<prosody>	Прерывание сессии	<disconnect>
	<phoneme>		<exit>
	<say-as>	Отправка и получение данных	<submit>
	<sub>	Исправление ошибок	<log>
	<mark>	Обработка событий	<catch>
	<s>		<error>
	<sentence>		<help>
	<p>		<noinput>
	<paragraph>		<nomatch>

Таблица 2

Описание тэгов VoiceXML

Тэг	Описание
<assign>	Устанавливает значение переменной
<audio>	Проигрывает аудиозапись пользователю
<block>	Содержит исполняемый код (не интерактивный)
<break>	Элемент SSML, вставляет паузу в выходной аудиопоток
<catch>	Перехватывает событие
<choice>	Определяет элемент меню
<clear>	Очищает одну или несколько переменных формы
<disconnect>	Прекращает телефонную сессию
<else>	Отмечает начало ИНАЧЕ-части внутри элемента <if>
<elseif>	Отмечает начало ИНАЧЕ_ЕСЛИ-части внутри элемента <if>
<emphasis>	Элемент SSML, меняет ударения в речевом выходе
<enumerate>	Генерирует аудиовыход, в котором перечисляются все пункты меню
<error>	Перехватывает ошибки
<example>	Фраза-пример, которая соответствует правилам грамматики
<exit>	Выход из сессии
<field>	Объявляет поле ввода в форме
<filled>	Содержит действия, которые должны выполниться после заполнения формы
<form>	Предоставляет информацию и собирает данные
<goto>	Переход
<grammar>	Определяет грамматику распознавания речи
<help>	Помощь
<if>	Оператор условия
<initial>	Объявляет начальную логику в момент входа в форму
<item>	Элемент XML-грамматики входа, указывает опциональную или повторяющуюся информацию от пользователя
<lexicon>	Элемент XML-грамматики входа, указывает источник информации о произношении
<link>	Указывает на общий для всех диалогов переход
<log>	Записывает информацию по отладке в журнал звонка
<menu>	Позволяет пользователю выбрать один из вариантов
<meta>	Определяет метаданные парой имя-значение
<noinput>	Перехватывает события отсутствия ввода
<nomatch>	Перехватывает события несовпадения
<object>	Всегда выбрасывает исключение о неподдерживаемом объекте
<one-of>	Элемент XML-грамматики входа, указывает на альтернативные данные от пользователя
<option>	Определяет опцию в <field>
<p>, <paragraph>	Элемент SSML, определяет параграфы в тексте
<param>	Определяет параметры в элементе <subdialog>
<phoneme>	Элемент SSML, определяет фонетическое произношение
<prompt>	Отправляет текст в очередь на генерацию звукового выхода
<prosody>	Элемент SSML, меняет речевой выход
<record>	Записывает аудиофрагмент
<reprompt>	Повторяет запрос в случае повторного посещения поля после события
<return>	Возвращает из поддиалога
<rule>	Элемент XML-грамматики входа, определяет правило грамматики

<ruleref>	Элемент XML-грамматики входа, определяет отношение к другому правилу грамматики
<s>, <sentence>	Элемент SSML, определяет часть текста как предложение
<say-as>	Элемент SSML, определяет, как должно быть произнесено слово
<script>	Указывает блок Javascript в клиентской части
<speak>	Ключевой элемент для отдельного SSML-документа
<sub>	Атрибуты этого тэга предоставляют альтернативный текст
<subdialog>	Обращение к другому диалогу как поддиалогу текущего
<submit>	Отправляет значения на сервер
<tag>	Элемент XML-грамматики входа, указывает, как интерпретировать ввод пользователя
<throw>	Создаёт событие
<token>	Элемент XML-грамматики входа, указывает, какие слова будут произнесены пользователем
<transfer>	Переадресует звонок
<value>	Вставляет значение выражения в выходной аудиопоток
<var>	Объявляет переменные
<voice>	Элемент SSML, запрашивает изменения в голосе
<vxml>	Содержит VoiceXML-код

Рассмотрим классический пример базовой программы. Сначала необходимо указать стандартный заголовок, с которого начинаются все документы VoiceXML:

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

Затем необходимо указать версию конкретного документа. В случае если вы хотите использовать функциональность самой новой версии 2.1 (например, элемент <data>), надо указать конкретную версию вашей программы: «2.1». Таким образом, получаем код:

```
<?xml version=«1.0» encoding=«UTF-8»?>
<vxml version = «2.1» >
  <form>
    <block>
    <prompt>
      Здравствуй, мир!
    </prompt>
    </block>
  </form>
</vxml>
```

Обратите внимание, что даже самый простой сценарий требует структуры кода. Тэг <form> отличается от аналогичного в HTML тем, что группирует секции ввода и вывода на вашей странице (необходимо помнить, что это «веб-страница» для телефона). В более сложных сценариях этот тэг может являться поименованной секцией, к которой можно перейти. Аналогично, <block> в нашем примере кажется чем-то неважным, потому что мы выполняем лишь одну функцию. Тем не менее, VoiceXML требует структуру внутри <form>, что и обеспечивает нам тэг <block>. Но как же VoiceXML сможет конвертировать «Здравствуй, мир»? Никаких дополнительных тэгов не надо, VoiceXML воспринимает любое значение, не ограниченное тэгами, как текст, который должен быть прочитан по телефону.

В VoiceXML определены два типа диалогов, которые позволяют взаимодействовать приложению с пользователем: формы и меню. Форма используется для представления или получения информации от пользователя. Меню является специализированной формой, которая требует сделать определённый выбор. Элемент **<form>** включает в себя директиву **<goto>**, которая говорит приложению, куда двигаться, в зависимости от пользовательского входа. Меню

используует элементы **<choice>** для определения дальнейшего пути на основе выбора пользователя. Следующий пример предлагает пользователю сделать выбор и вызывает различные файлы в зависимости от выбора.

```
<?xml version=«1.0» encoding=«UTF-8»?>
    <vxml version = «2.1»>
    <form id=«F1» scope=«document»>
     <catch event=«Event_1 Event_2»>
     <prompt>
       <value expr=«_message»/>
     </prompt>
     </catch>
     <menu id=«menu»>
     <prompt>
    Что такое НАНБ?
       Если вы думаете, что это Национальная академия наук Беларуси,
    скажите слово «академия» или нажмите «один».
      Если вы думаете, что это фрукт, скажите слово «фрукт» или нажмите «два».
     </prompt>
     <choice event=«Event_1» accept=«approximate»
          dtmf=«1»
          message=«Верно!»>
           наверное, это академия
     </choice>
     <choice event=«Event_2» accept=«exact»
          dtmf=«2»
          message=«Неправильно»>
           фрукт
     </choice>
     </menu>
    </form>
    </vxml>
```

Кроме прочего, можно обратить внимание на параметр accept тэга <choice>: у него есть значения exact и approximate. Exact требует полного совпадения, а approximate допускает совпадение даже по одному слову, так что в нашем примере, распознав слово Academy, программа признает выполненным условие «most definitely intended to be an Academy».

В данном примере были использованы всего семь элементов:

```
<vxml>
<form>
<catch>
<prompt>
<value>
<menu>
<choice>
```

Конечно, здесь не были описаны все возможности VoiceXML, но приведённые примеры могут помочь составить представление о том, как голосовые команды обрабатываются сервером. Конечно, такой тип приложений должен

учитывать масштабируемость и нагрузку на сервер, а традиционные серверные задания (доступ к базе данных, система сообщений, запросы и пр.) должны разрабатываться в виде строк выбора.

4. Возможности применения

Существует много уже реализованных применений VoiceXML.

1) Интерактивный голосовой ответ (IVR) и расширенное самообслуживание. Такие решения автоматизируют звонки, что позволяет повысить качество обслуживания, снизить затраты, сократить сроки возврата инвестиций и продвигать новые прибыльные возможности.

2) Контактный центр. Системы управления взаимоотношения с клиентами (CRM) — это клиенто-ориентированные стратегии бизнеса, целью которых является оптимизация прибыльности, оборотных средств и удовлетворения потребностей клиентов. Хотя цели во всех организациях одинаковы, реализация конкретных решений всегда уникальна.

3) Унифицированные коммуникации. К таким системам относится широкий спектр приложений, разработанных для преодоления барьера между разными типами связи: телефон, электронная почта, чат, веб и др. Многие компании инвестируют в развитие таких технологий для дальнейшего их внедрения в CRM-приложения. Наличие интегрированной в CRM-приложение кнопки «нажмите для звонка» предотвращает ошибки ручного набора и делает работу агентов более продуктивной. Такое приложение с интегрированными телефонными коммуникациями считывает номер, с которого производится звонок, и проверяет CRM-систему на совпадения. В случае если номер принадлежит клиенту, у агента открываются соответствующие записи. Это позволяет агентам приветствовать звонящих по имени и более продуктивно вести диалог, что приводит к сокращению времени разговора. С такими решениями среднее время каждого звонка снижается на 20–60 секунд.

4) Уведомления и напоминания. Автоматические мультимодальные системы уведомлений гарантируют, что клиенты, работники и другие заинтересованные стороны получат срочные сообщения, даже если их не ожидают. Уведомления и напоминания — развивающийся важный бизнес, который повышает лояльность клиентов и открывает возможности для увеличения прибылей. VoiceXML предоставляет следующие функциональные возможности:

a) обработка входящих и исходящих звонков;
b) отправка персонализированных сообщений на основе систем синтеза речи;
c) конференц-связь;
d) возврат звонка;
e) факс;
f) динамические меню;
g) наблюдение и составление отчётов;
h) облегчённая интеграция с вебом и электронной почтой.

Заключение

На данный момент VoiceXML является общепринятым языком разметки для голосовых приложений. Этому способствуют как функциональные факторы: простота, логичность, хорошее описание, доступность, — так и организационные: язык разрабатывается при поддержке W3C [14] и крупнейших мировых компаний — лидеров IT-отрасли.

Литература

1. VoiceXML Forum, http://www.voicexml.org/.
2. Ericsson, http://www.ericsson.com/.
3. Siemens AG, http://w1.siemens.com/.
4. NOVELL, http://www.novell.com/.
5. France Telecom, http://www.francetelecom.com/.
6. SAMSUNG, http://www.samsung.com/.
7. Sun Microsystems, http://www.sun.com/.
8. IBM, http://www.ibm.com/.
9. AT&T, http://www.att.com/.
10. Lucent Technologies, http://www.alcatel-lucent.com/.
11. Motorola, http://www.motorola.com/.
12. VoiceXML's History, Luann Martinez, 2009, http://www.voicexml.org/voicexml-tutorials/introduction/voicexmls-history/.
13. Harry Newton, Newton's Telecom Dictionary, 24th Edition: Telecommunications, Networking, Information Technologies, The Internet, Flatiron Publishing, 2008.
14. W3C — The World Wide Web Consortium, http://www.w3.org/.
15. Voice Extensible Markup Language (VoiceXML) 2.1, http://www.w3.org/TR/voicexml21/.
16. Speech Synthesis Markup Language (SSML) Version 1.0, http://www.w3.org/TR/speech-synthesis/.
17. Speech Recognition Grammar Specification Version (SRGS) 1.0, http://www.w3.org/TR/speech-grammar/.
18. Salmon, Backus-Naur Forms, Irwin Professional Publishing, 1992.
19. Semantic Interpretation for Speech Recognition (SISR) Version 1.0, http://www.w3.org/TR/semantic-interpretation/.
20. IBM WebSphere Voice, http://www-01.ibm.com/software/voice/.
21. Microsoft Speech Technologies, http://www.microsoft.com/speech/speech2007/default.mspx.
22. Speechtech s.r.o., http://www.speechtech.cz/.
23. Voxeo IVR Platforms, http://www.voxeo.com/.

Е.Б. Никитин —

окончил факультет прикладной математики Белорусского государственного университета, прошел инструкторские курсы Cisco CCNA в Киевском национальном университете имени Тараса Шевченко, аспирант дневной формы ОИПИ НАН Беларуси.