Chapter 3

In Chapter 2, "VoiceXML essentials," on page 24 our exploration of VoiceXML was motivated by how the language is generally used. This chapter will exhaustively cover the entirety of VoiceXML from the perspective of its Document Type Definition (DTD). We will begin with a brief overview of the structure of the language, defining several abstract element types that will be used as a shorthand throughout the remainder of this chapter as we examine each language element type individually.

This chapter uses the graphical Unified Modeling Language (UML) to visualize the VoiceXML elements and their relationships. Appendix F, "UML Class Diagram primer," on page 446 briefly summarizes UML.

This chapter indicates attributes based on VoiceXML 2.0 Working Draft. New attributes that are exclusive to VoiceXML 2.0 implementations — i.e. not available in VoiceXML 1.0 implementations — have the notation "(2.0)".

VoiceXML document structure

Each VoiceXML document must have a `vxml` element as its root node. Figure 3–1 lists the legal children of the `vxml` element type.

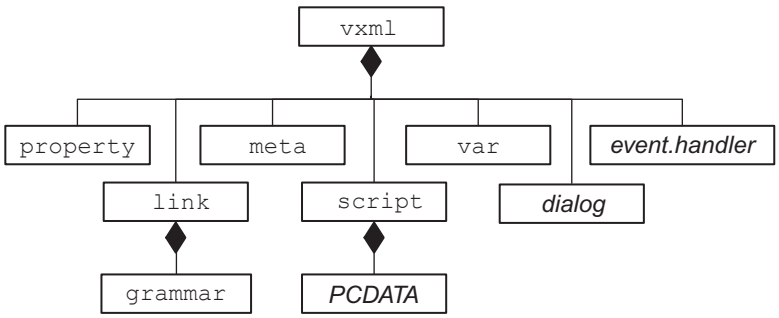

Figure 3–1 VXML element type's legal children

All of these children are other VoiceXML element types except two: `event.handler` and `dialog`. These are abstract element types, defined as entities in the DTD. Each of them can represent a union of element types.

Figure 3–2 shows the `event.handler` element types. Any of these `event.handler` element types can legally be a child of the `vxml` element type.

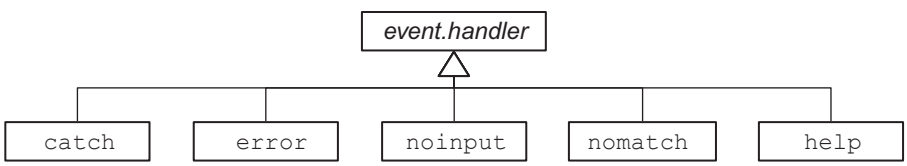

Figure 3–2 The `event.handler` element types

Figure 3–3 shows the definition of the abstract `dialog` element type which includes `form` and `menu`. These are the two types of dialogs defined in VoiceXML. Both of these element types may be children of a `vxml` element.

The `menu` element type is the parent of `choice`, `property`, or `prompt` element types as well as `event.handler` element types, previously defined in Figure 3–2, and `audible` element types which will be defined shortly. Similarly, the `form` element type is the parent of

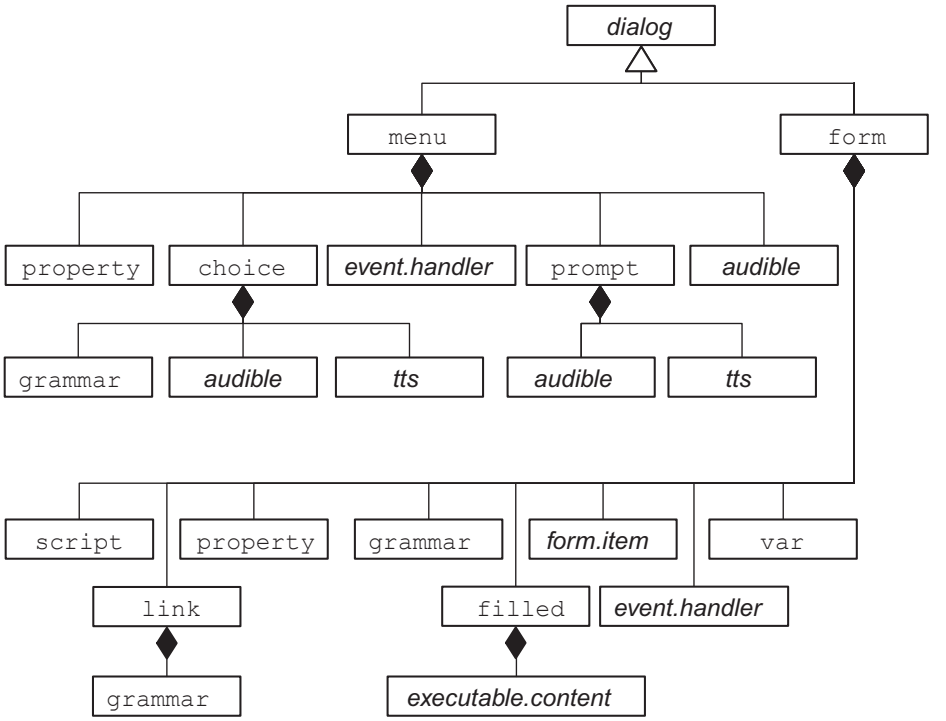

Figure 3–3 The `form` and `menu` dialog element types

`script`, `property`, `link`, `filled`, `var`, and `grammar` element types as well as `event.handler` element types, previously defined, and `form.item` element types.

The `form.item` elements, shown in Figure 3–4, play a particular role in the Form Interpretation Algorithm.

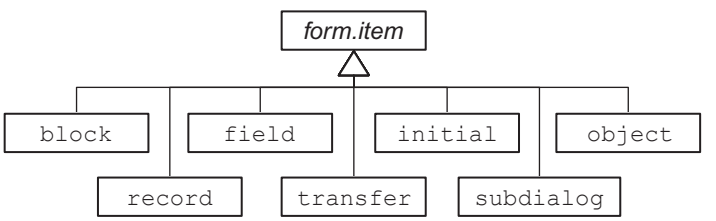

Figure 3–4 The `form.item` element types

The element types that each form item can contain are covered in more detail in the respective element type sections below.

Another abstract element type that appears in Figure 3–3 is `executable.content`. Executable content elements are typically found as children of either the `block`, `filled`, `if`, or `event.handler` elements. Figure 3–5 shows the element types that can be interpreted as `executable.content`. The VoiceXML interpreter typically treats such elements as procedural statements.

The element types that are children of the `executable.content` elements are discussed in the respective following sections. It is worth pointing out that the abstract element type `variableAccessor` shown in Figure 3–5 is not actually in the DTD but is added here for clarity.

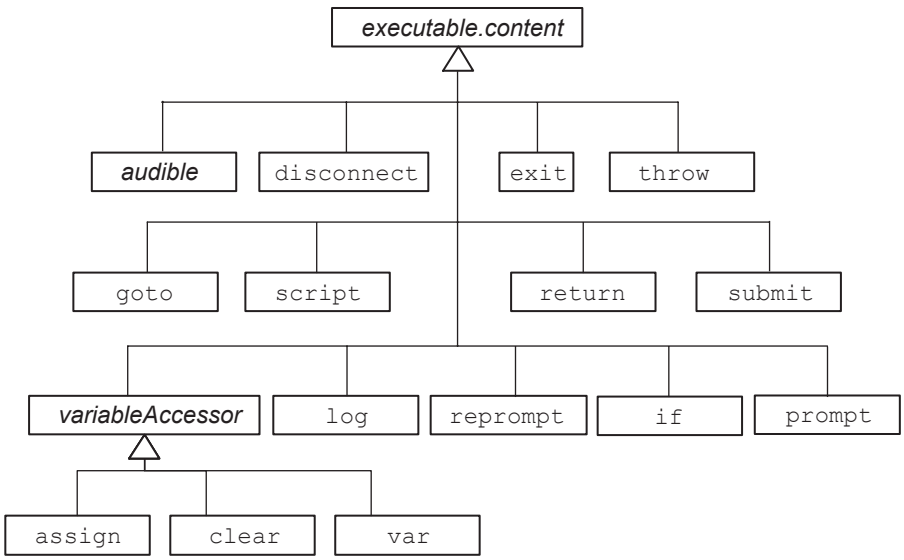

Figure 3–5 The `executable.content` element types

The `audible` element types represent audio that can be played back. It is worth noting that `audible` is implemented in the DTD as an entity named `audio`. Since there is also a VoiceXML element type `audio`, we will refer to this entity as `audible` in this text for the sake

of disambiguation. Figure 3–6 lists the `audible` element types. You will notice that `PCDATA` (parsed character data) can be interpreted as `audible` content, as is the case with plain text processed by a text-to-speech processor.

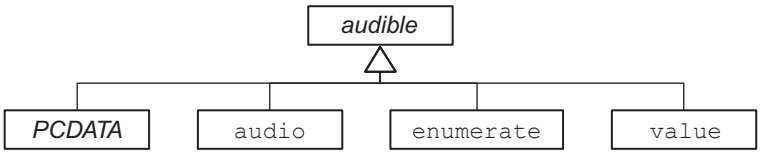

Figure 3–6 The `audible` element types

Looking back at Figure 3–3 we see that the `menu`, `choice`, and `prompt` element types all have `audible` element types as their children.

In addition to `audible` element types, there is also the abstract `tts` element type, referring to Text-To-Speech. The element types that can be interpreted as `tts` are shown in Figure 3–7.

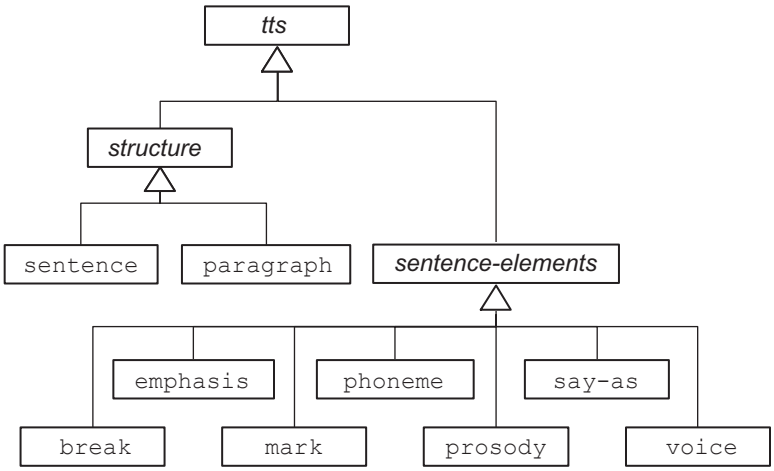

Figure 3–7 The `tts` element types

The `tts` element types are new to VoiceXML 2.0 and are used to either organize or mark up text to be processed by a text-to-speech processor. As we can see from Figure 3–3, `choice` and `prompt` element types can be parents of `tts` element types.

While the element types for controlling audio output, namely `audible` and `tts`, can be found as children of various VoiceXML element types, the element types for controlling audio input are found only as children of the `grammar` element type. Figure 3–8 shows the taxonomy of a `grammar` definition using the GRXML standard grammar definition format.

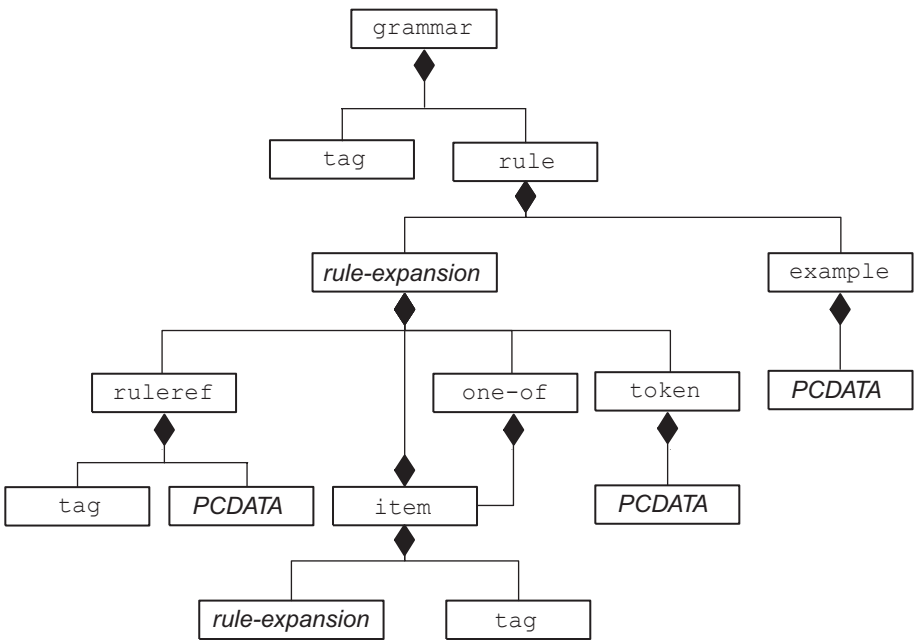

Figure 3–8 The `grammar` element type taxonomy

The details of each of these element types will be described in the sections that follow.

Attribute types

VoiceXML element attributes are in many cases type-restricted by the VoiceXML DTD. Some of the more common attribute types are summarized below:

`NMTOKEN`
An XML name token (see *The XML Handbook*).

`CDATA`
Character data to be interpreted verbatim (see *The XML Handbook*).

`PCDATA`
Parsed character data (see *The XML Handbook*).

`ID`
A unique element identifier (see *The XML Handbook*).

`bargeintype`
The enumeration (`energy | speech | recognition`) used to indicate what constitutes a barge-in event.

`boolean`
The enumeration (`true | false`).

`content.type`
`CDATA` containing a MIME (Multipurpose Internet Mail Extensions) content-type identifier.

`duration`
`CDATA` containing a time duration written as a number followed by a unit abbreviation (`s`, `ms`, etc.).

`event.name`
An `NMTOKEN` used to name a particular event type.

event.names
CDATA comprising a space-delimited list of event.names.

expression
CDATA representing an ECMAScript expression.

field.name
An NMTOKEN used to name a form field.

field.names
CDATA comprising a space-delimited list of field.names.

phoneme-alphabet
CDATA containing the name of a phoneme alphabet.

phoneme-string
CDATA containing a word spelled in a phoneme alphabet.

scope
An enumeration (document | dialog) used to indicate the scope of a grammar rule.

uri
CDATA representing a Uniform Resource Identifier (URI).

voice-name
CDATA containing the name of a text-to-speech voice "personality."

voice-names
CDATA comprising a space-delimited list of voice-names.

Element categories

We have explored the taxonomy of a VoiceXML document. The VoiceXML elements can also be categorized by function, as shown in Table 3–1, based on comments in the DTD. This partitioning can be summarized as follows:

Audio Input
Element types controlling how input is collected from the caller.

Audio Output
Element types controlling how audio is rendered to the caller.

Call Control
Element types that perform telephony hardware operations.

Dialog
Element types that define a dialog.

Event
Element types that produce or handle real-time events.

Field
Element types that define a form field.

Flow Control
Element types that control the execution flow of a VoiceXML application.

Miscellaneous
Element types that provide access to ECMAScript, native objects, and run-time logging.

Prompt
Element types controlling the prompting of the caller.

Root
Element types used to declare a VoiceXML document and provide document-level metadata.