

web server **202**. As discussed below, web server **202** can also include an authoring mechanism that can dynamically generate client-side markups and scripts. In a further embodiment, the web server **202**, recognition server **204** and client **30** may be combined depending on the capabilities of the implementing machines. For instance, if the client comprises a general purpose computer, e.g. a personal computer, the client may include the recognition server **204**. Likewise, if desired, the web server **202** and recognition server **204** can be incorporated into a single machine.

[0044] Access to web server **202** through phone **80** includes connection of phone **80** to a wired or wireless telephone network **208**, that in turn, connects phone **80** to a third party gateway **210**. Gateway **210** connects phone **80** to a telephony voice browser **212**. Telephone voice browser **212** includes a media server **214** that provides a telephony interface and a voice browser **216**. Like device **30**, telephony voice browser **212** receives HTML scripts or the like from web server **202**. In one embodiment, the HTML scripts are of the form similar to HTML scripts provided to device **30**. In this manner, web server **202** need not support device **30** and phone **80** separately, or even support standard GUI clients separately. Rather, a common markup language can be used. In addition, like device **30**, voice recognition from audible signals transmitted by phone **80** are provided from voice browser **216** to recognition server **204**, either through the network **205**, or through a dedicated line **207**, for example, using TCP/IP. Web server **202**, recognition server **204** and telephone voice browser **212** can be embodied in any suitable computing environment such as the general purpose desktop computer illustrated in **FIG. 3**.

[0045] However, it should be noted that if DTMF recognition is employed, this form of recognition would generally be performed at the media server **214**, rather than at the recognition server **204**. In other words, the DTMF grammar would be used by the media server **214**.

[0046] Referring back to **FIG. 4**, web server **202** can include a server side plug-in authoring tool or module **209** (e.g. ASP, ASP+, ASP.Net by Microsoft Corporation, JSP, Javabeans, or the like). Server side plug-in module **209** can dynamically generate client-side markups and even a specific form of markup for the type of client accessing the web server **202**. The client information can be provided to the web server **202** upon initial establishment of the client/server relationship, or the web server **202** can include modules or routines to detect the capabilities of the client device. In this manner, server side plug-in module **209** can generate a client side markup for each of the voice recognition scenarios, i.e. voice only through phone **80** or multimodal for device **30**. By using a consistent client side model, application authoring for many different clients is significantly easier.

[0047] In addition to dynamically generating client side markups, high-level dialog modules, discussed below, can be implemented as a server-side control stored in store **211** for use by developers in application authoring. In general, the high-level dialog modules **211** would generate dynamically client-side markup and script in both voice-only and multimodal scenarios based on parameters specified by developers. The high-level dialog modules **211** can include parameters to generate client-side markups to fit the developers' needs.

Exemplary Client Side Extensions

[0048] Before describing dynamic generation of client-side markups to which the present invention is directed, it may be helpful to first discuss an exemplary form of extensions to the markup language for use in web based recognition.

[0049] As indicated above, the markup languages such as HTML, XHTML, cHTML, XML, WML or any other SGML-derived markup, which are used for interaction between the web server **202** and the client device **30**, are extended to include controls and/or objects that provide recognition in a client/server architecture. Generally, controls and/or objects can include one or more of the following functions: recognizer controls and/or objects for recognizer configuration, recognizer execution and/or post-processing; synthesizer controls and/or objects for synthesizer configuration and prompt playing; grammar controls and/or objects for specifying input grammar resources; and/or binding controls and/or objects for processing recognition results. The extensions are designed to be a lightweight markup layer, which adds the power of an audible, visual, handwriting, etc. interface to existing markup languages. As such, the extensions can remain independent of: the high-level page in which they are contained, e.g. HTML; the low-level formats which the extensions used to refer to linguistic resources, e.g. the text-to-speech and grammar formats; and the individual properties of the recognition and speech synthesis platforms used in the recognition server **204**. Although speech recognition will be discussed below, it should be understood that the techniques, tags and server side controls described hereinafter can be similarly applied in handwriting recognition, gesture recognition and image recognition.

[0050] In the exemplary embodiment, the extensions (also commonly known as "tags") are a small set of XML elements, with associated attributes and DOM object properties, events and methods, which may be used in conjunction with a source markup document to apply a recognition and/or audible prompting interface, DTMF or call control to a source page. The extensions' formalities and semantics are independent of the nature of the source document, so the extensions can be used equally effectively within HTML, XHTML, cHTML, XML, WML, or with any other SGML-derived markup. The extensions follow the document object model wherein new functional objects or elements, which can be hierarchical, are provided. Each of the elements are discussed in detail in the Appendix, but generally the elements can include attributes, properties, methods, events and/or other "child" elements.

[0051] At this point, it should also be noted that the extensions may be interpreted in two different "modes" according to the capabilities of the device upon which the browser is being executed on. In a first mode, "object mode", the full capabilities are available. The programmatic manipulation of the extensions by an application is performed by whatever mechanisms are enabled by the browser on the device, e.g. a JScript interpreter in an XHTML browser, or a WMLScript interpreter in a WML browser. For this reason, only a small set of core properties and methods of the extensions need to be defined, and these manipulated by whatever programmatic mechanisms exist on the device or client side. The object mode provides eventing and scripting and can offer greater functionality to give the