Pervasive solutions
White paper

Mainstreaming speech-enabled Web applications.

By Juan Huerta, David Lubensky, David Nahamoo, Roberto Pieraccini, T. V. Raman and Charlie Wiecha, IBM T.J. Watson Research Center

September 2004

Contents

Introduction

Over the past decade, the Internet has evolved from a repository of static HTML pages toward dynamically generated Web user interfaces built on standardized formats and protocols. This evolution has been accelerated by the adoption of standardized Web programming models that have driven overall cost out of application deployment. The graphical user interfaces (GUIs) users encounter when they access Web applications constitute the presentation layer, or *view*, of the visual Web. But there is an underlying Web, including scripts, Java™ programs, servlets and portlets that is at the core of this visual Web.

The past decade has also seen the maturation of the speech-technology industry. Speech is a powerful tool, capable of delivering a natural way to access information and services to millions of end users. Deploying speech technologies on the Web can help companies further lower the cost associated with customer care. Similar to visual Web applications, speech applications can act as a presentation layer, the *voice user interface* (VUI), that interacts with a common underlying application layer. Thus, if you add voice access to Web applications using a mainstream Web programming model, you can leverage your existing application infrastructure.

The unification of the visual and voice Web programming model can be referred to as *speech-enabling the Web*. This doesn't mean a one-to-one translation of the visual presentation. Speech applications require more robust error control and sophisticated dialog management. However, most of the logic and data layers of Web applications can be reused when providing voice access.

This white paper outlines the evolution of speech application development from today's static content to dynamically generated VoiceXML. As developers gain access to interoperable voice components within Web programming models, speech specialists can package best-of-breed VUIs that can be used to speech-enable Web applications. With this technology, companies can more cost-effectively add speech to Web applications—without sacrificing the quality of the resulting VUI. Reusable Dialog Components (RDCs), a component framework based on JavaServer Pages (JSP), is central to this evolution. IBM is taking the lead in this evolution by donating an open-source implementation of the RDC framework to Apache.

This white paper defines the roadmap for driving overall cost out of creating, deploying and managing speech solutions. It also discusses how complex speech applications built with today's technologies can interoperate with speech-enabled Web applications to help ensure a smooth transition and provide a seamless end-user experience.

Taking the right steps forward

The speech-technology industry has taken its first step toward the adoption of a Web programming model by standardizing on VoiceXML, Version 2.0. First-generation voice-enabled Web applications were mostly built of static VoiceXML pages. The next step is a move to complex applications deployed on standard Web servers and implemented through programs that deliver dynamically generated VoiceXML markup. Just like the visual Web, the next step in mainstreaming speech-enabled Web applications is to adopt uniform programming models to create and deploy these speech-enabled Web applications. On the visual Web, creating sophisticated user interaction is mediated by component libraries that ease the generation of complex HTML pages. The move to dynamically generated VoiceXML requires similar component libraries that capture best practices in VUI design.

Mainstreaming voice access to the Web changes today's practice of developing entire speech applications to a model where voice access is achieved by replacing the visual view layer with a high-quality VUI. In this model, Web applications are developed using standard application frameworks such as *Struts*; voice access is achieved by creating appropriate views that are assembled from a set of reusable and configurable components. Such components need to be created within a framework that encourages interoperability across components to help unify the speech applications market.

IBM is spearheading this effort with the initial donation of a set of reference RDCs and the supporting framework integrated within the Java 2 Platform, Enterprise Edition (J2EE) and JSP programming models. The reference implementation of this framework is available as open source through the Apache Jakarta Taglibs project. The inital set of components and the underlying framework is the beginning of a community effort toward the evolution of a common programming model for voice interaction based on J2EE and JSP technology. With this framework, developers can avoid compartmentalizing speech into its own application-development niche. Members of the community supporting this initiative can use the framework and reference implementation according to their business models. They can also contribute to extending the framework, as well as to increasing the catalog of available components. The speech-technology industry can continue to provide specialized knowledge and research that is delivered to mainstream Web developers in the form of sophisticated components.

Components designed to give speech-specific technology to Web developers

Unifying the visual and voice Web can lead to a common framework that consists of collecting and presenting information. Visual Web applications perform user interaction through widgets assembled on HTML pages. These buttons, check boxes, fillable text fields, scrollable lists, and other data gathering and presentation mechanisms are the components that make up the visual Web. Specialized components that deal with specific types of data, such as money, time, dates and addresses, help reduce the cost of implementing complex interactive applications. The use of such component frameworks can greatly accelerate the development of visual Web applications through the process of customization and reuse.

Whereas the visual Web can rely on a persistent visual display backed by error-free user input, the speech medium is temporal and nonpersistent. Speech interaction is characterized by a sequence of turns where requests or pieces of information are alternatively spoken by the system and by the user. Although it is advancing at a fast pace, speech-recognition technology is still error-prone and needs to be backed up by confirmation, correction and reprompting. With prepackaged dialog components, Web developers could more-efficiently handle these aspects of conversational interaction, and ease the overall task of speech enablement.

To be used effectively by nonspeech specialists, speech components must embed much of the specific knowledge that enables the creation of high-quality speech interfaces. Thus, you must incorporate grammars, prompts, confirmation and correction strategies into these components. You must also ensure that the components are sufficiently configurable to allow reuse within a wide range of applications. Finally, you should be able to put together sophisticated components from simpler ones.

The RDC framework embodies all of these features. RDCs are interoperable components within the J2EE and JSP framework that offer a means to bring speech-specific knowledge to Web developers. Each RDC component is composed of a data model, speech-specific assets like grammar and prompts, configuration files, and the dialog logic needed to collect a piece of information. The VoiceXML that performs the VUI is generated by the component implementation. A developer writes an application by instantiating these components and specifying their run-time behaviors through component attributes and configuration files. The data model is where components store the values collected from the user interaction; components handle data validation and normalization. Component data models are implemented as Java beans. Each component implements a set of tasks including data collection, confirmation, validation and disambiguation. Component authors can provide custom implementations for all of these tasks.

Atomic RDCs collect simple data values such as a time, the name of a place or an alphanumeric string; atoms can be put together to form composite RDCs. Composite and atomic RDCs can be aggregated to form more complex components. The resulting composite RDCs are structured in the same way as atomic ones. They also have a data model, implement sets of tasks and include speech-specific assets. Their behavior is specified by attributes and configuration files. The framework provides a container tag to facilitate the construction of composite RDCs. The container implementation invokes a pluggable dialog-management strategy that controls the activation of the constituent RDCs. The framework provides a default-directed dialog strategy that a developer can override.

RDCs, as envisioned in this white paper, must be developed within a community process to help ensure interoperability among components. A first step in the creation of such a community is IBM's donation to apache.org of an open-source reference implementation of the RDC framework.

Taking cost out of developing speech solutions

Building on standardized programming models creates the opportunity for developing mainstream tools for speech-enablement. This section outlines the roadmap for how IBM sees today's world of speech-oriented applications evolving toward a world where speech-enablement is just another aspect of overall application development.

Adopting the Web programming model for voice interaction

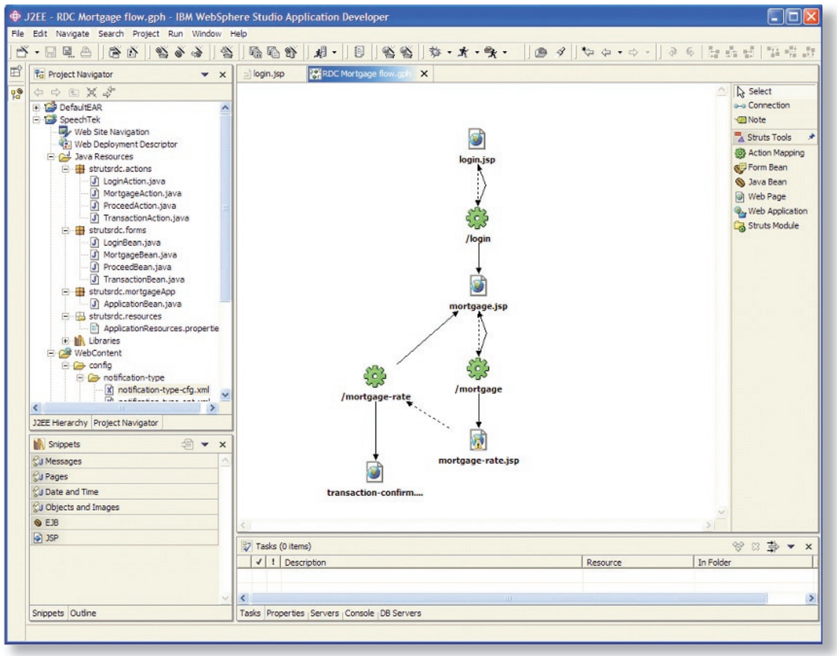
The speech-technology industry took one of its first steps toward integrating voice interaction with mainstream applications with its adoption of VoiceXML and the associated Web programming model built around HTTP and distributed resources that are identified through URLs. This has allowed the speech-technology industry to move away from speech applications that are written as executable programs that link directly to the underlying speech engines. Today, voice applications can be developed using standards-compliant VoiceXML, Version 2.0 that avoids tying the final application to any specific vendor's engine application programming interfaces (APIs).

From static to dynamic VoiceXML

The next step in this evolution is to create Web applications that emit standards-compliant VoiceXML. This follows the same evolutionary pattern as seen on the visual Web; static HTML pages have been replaced over time by server-side Web application frameworks that emit HTML. Server-side deployment of Web applications has been facilitated by the creation of standardized Web programming models that abstract the details of back-end integration, as well as the underlying business logic that determines the transitions among different stages in an application. This helps Web developers integrate user tasks into ever-larger applications. As the speech-enabled Web evolves in an analogous manner, voice application development moves from today's voice-specific programming model and associated tools to one where voice interaction is authored as a specialized view that binds to a common underlying Web application.

Tools for voice enablement

Tools for speech-enabling Web applications can integrate seamlessly with mainstream Web application tools. An example is the Struts builder available within the IBM WebSphere® Studio Application Developer tool, as shown in the figure below. This will also allow speech specialists to focus on the task of creating high-quality voice user interaction without having to develop the complete application. These VUI components can incorporate best practices of VUI design and help ensure that speech-enabling Web applications are not created by sacrificing the quality of the user experience. Finally, during this transition period, existing speech-enabled applications created within today's voice-centric programming models can still be integrated into the overall application flow by using the underlying Web framework defined by HTTP. As an example, a voice-enabled financial portal created by binding a VUI to an underlying Web application might choose to invoke a pre-existing speech-bank application through a URL—or more generally, as a Web service.



Struts allows the separation of the presentation layer from the underlying application flow. Struts applications can be voice-enabled by replacing visual-view JSP pages with RDC-based JSP pages that produce the voice view.

The goal: Drive cost out of voice applications

When the transition to speech-enabling Web applications is complete, IBM expects the overall cost of voice-enablement to be significantly reduced from today's levels, because each link in the overall end-to-end value chain of speech application deployment can focus on a specific core competency.

Value propositions and business opportunities

This section outlines how the mainstreaming of speech solutions by adding speech-enablement to the overall portfolio of Web technologies creates new business opportunities for different segments of the speech industry. The end-to-end value chain that makes up the creation, deployment and delivery of voice applications comprises several parts. At present, vendors play in more than one part of this value chain—some of them in at least two or three neighboring sectors. IBM's long-term goal is to help each class of vendors focus on their particular core competencies, while relying on interoperability that comes from using standards.

Voice platform vendors

The momentum behind VoiceXML, Version 2.0 has created an exponential growth in the software industry, and IBM expects this trend to be enhanced by the ability to speech-enable J2EE Web applications using a standardized programming model that provides robust access, while controlling overall total cost of ownership (TCO). The ability of the mainstream Web programmer to generate high-quality VUIs expressed in VoiceXML can significantly enhance the value of robust VoiceXML browsers.

Hosting

A standardized deployment environment based on the widely used and tested J2EE Web application architecture helps control the overall cost of hosting and maintaining speech-enabled applications.

Speech-recognition and text-to-speech (TTS) engines
J2EE Web developers can leverage the evolution of speech technologies to deliver on demand spoken access to Web services. This can create more volume of the market request for speech technology, which can become part of the standard assets for Web applications. Engine vendors can be enticed to add advanced functionality and technological improvement to their core technologies to support advanced requirements defined by component creators and Web developers.

Development tools
Having tools that are consistent with interoperable components helps create libraries of speech-enabling building blocks. This can lead to rapid application development(RAD)—freeing developers to focus on more-sophisticated user interactions.

Enterprises and service providers
The ability to speech-enable standard J2EE Web applications using the widely available skill set of J2EE and JSP Web development can bring spoken access to businesses quickly and cost-effectively—helping to control TCO.

Application developers
Having a standardized Web-programming model and associated tooling to create dynamic voice access to Web applications and services helps reduce the cost of developing on demand voice-enabled solutions. Speech-enabling J2EE applications through JSP technology, using dialog components, can create demand for application development services based on this standard programming model.

Conclusion

Speech-recognition technology is mature, and mainstream deployment of speech solutions can drive down costs in key areas such as customer care. Reducing the cost of creating, managing and deploying mainstream speech applications requires that developers build on standardized Web-programming models. This can turn speech-enablement into yet another access channel to mainstream Web applications. Enabling this evolution without sacrificing the overall quality of the end-user experience requires the packaging of speech-interaction expertise into standardized components that can be integrated into mainstream Web development environments. IBM plays a key role in this evolution through the Apache community process by contributing an open-source reference implementation of an RDC framework. This framework builds on the success of VoiceXML and J2EE and helps in creating voice-enabled J2EE Web applications.

For more information

To learn more about IBM speech technology, visit:

ibm.com/pervasive

G224-9114-00