

VoiceObjects

all business is vocal

The Case for Voice Application Management Systems

A White Paper prepared by VoiceObjects

July 2005

Table of content

Executive Summary 4

Introduction to Technology 5

Life Cycle of Voice Applications 7

Voice Application Fundamentals 9

Development is Internet Centric 9

Requirements for Voice Application Development 10

Voice Application Servers as Central Hub 11

Dynamic Dialog Generation 12

Continuous Dialog Analysis & Maintenance 14

Dynamic Personalized Voice Applications 15

Calling the Right Architecture 16

Voice Application Servers need to Support Standards 16

Voice Application Servers need to Scale 17

Developers need to Scale too 18

Conclusion 19

The VoiceObjects Advantage 20

Glossary 21

Target Audience: Development Managers, IT experts, and Web Project Managers / Strategists with little speech processing experience who are

1) responsible for delivering state-of-the-art voice applications with flexible dialogs and personalization options, comparable to personalized dynamic web sites of today, and
2) facing high support and development costs from handling older proprietary telephony systems, or are developing new, hand-coded Java installations instead of a competitive feature rich voice applications.

This paper focuses on the components involved in voice projects and the changes in architectures and technology that have occurred over the past few years.

Executive Summary

In today's business environment customers expect access to corporate resources via both the telephone and the web. The challenge for IT and telecommunication professionals is to service both of these channels in a consistent, complete, and cost effective manner. Telephony service systems are now referred to as voice application systems and they should allow callers to interact intuitively, quickly, and flexibly with a company and its information systems just as they can do with Web applications.

In general these voice application systems need to a) support a convenient caller-driven dialog, b) intelligently integrate back-end information into the information flow, and c) support access to other information distribution channels such as SMS, email, fax, etc.

Strategic IT executives and Project Managers must choose between the two principal approaches to developing voice application systems:

- **Hand-crafted server libraries**

  This approach creates the need to learn languages such as VoiceXML and it lacks tools for expediting the application building process. This solution typically involves the use of non-voice-specific tools and embedding vendor-specific tags or APIs into the code. As a result there is significant manual work involved. Maintenance becomes an issue as applications become larger and integrate different external data. Usability suffers because dialog flow for different voice applications becomes inconsistent and the user experience may vary unnecessarily across applications.

- **Voice Application Management Systems**

  Process-driven management systems address the issues presented by hand-crafted systems. They combine the use of process modeling with the ability to tap directly into enterprise application business logic and data sources. These systems typically employ a voice specific integrated approach to create, test, deploy and analyze voice applications. Usability is greatly enhanced because process and dialog are developed specifically for the voice channel with best practice for speech dialogs built into the system. Companies in this category also provide integrated scalable Voice Application Servers and development tools, rather than just one or the other of those capabilities.

While both leverage existing web infrastructure, using VoiceXML as the standard, Voice Application Management Systems go a step further. They explicitly use a standards-based, managed server architecture for the creation of dynamic and personalized voice dialogs. Handcrafted library management, on the other hand, relies on the skill level of the respective programmer to overcome the obstacles of creating dynamic voice applications.

Introduction to Technology

Speech recognition, especially speaker independent technology, has evolved rapidly over the last five years. This was initially fuelled by advances in computer processing power and then by rapid progress in speech recognition algorithms. The speech recognition algorithms could be implemented as software programs that could process information quickly enough on standard servers to be truly "real time". Speech recognition systems of today are the embodiments of these new algorithms, and they are deployed in such applications as telephone directory assistance, voice dialing, email reading by phone, and customer self-service applications for call centers.

In addition the latest generation of speech synthesis technology or text-to-speech (TTS) uses new algorithms that intelligently assemble snippets of real human voice to deliver synthesized speech, without the robotic sounds of earlier generations.

Speech processing software today represents a very real option for user input with high recognition accuracy (in the 95% range) and good quality speech output gives an even more human like interaction. The speech processing industry has also proposed new standards to allow for better integration and interaction. The industry today uses VoiceXML[1] and in parts SALT[2] to provide standardized interfaces to such systems. The standards are embraced by nearly all manufacturers of Automated Speech Recognition (ASR) and Text-To- Speech (TTS) systems, as well as providers of Interactive Voice Response (IVR) systems and voice service providers (VSP). This reshaping of the industry took place over the last three years, and today allows the development of modern voice applications using these programming standards.

But building voice applications has its own special challenges, especially if it is compared to the more familiar development of web applications. The difference between a graphical user interface (GUI) like a web site and a voice user interface (VUI) is profound. The main difference is the restriction of the communication and information channel: It's sequential only. So building applications for such an interface is a complex endeavor. However, just having a voice application (or even an older IVR system) does not necessarily provide organizations with the business benefits of efficiency, cost reduction, etc. The key success factor for a voice application is broad usage and user acceptance. Neither comes without good planning. Voice as a communication channel requires new design techniques, interface styles, and raises special integration and implementation issues. Successful voice applications will incur other major IT challenges in terms of scalability, availability, and accessibility of such applications.

To fulfill the new requirements, IT and telecom organizations need to provide a framework that allows designers and developers to create new systems and easily integrate them with existing IT back-end services. These environments are referred to as Voice Application Man-

1) VoiceXML – Voice mark-up language maintained by W3C. De-facto standard for creating voice services.

2) SALT – Mark-up language with voice and multimodal focus. Initiated by the SALT forum

agement Systems (VAMS), employing Voice Application Servers and existing Web infrastructure, as discussed in the following sections.

Similar to web application servers that support large web portals, a scalable server architecture must be in place to support large voice applications for millions of parallel callers. Similar to existing infrastructure components for web applications such as Content Management Systems (CMS) or Portal Management Systems (PMS), a Voice Application Management System (VAMS) comprises of similar components for design, operation and maintenance. Figure 2 illustrates the claim for a Voice Application Server as the hub for such a complete VAMS. VoiceObjects provides an Internet-centric and standards-based Voice Application Management System that allows enterprises, carriers, and service providers to deliver voice-driven self-service and customer care solutions to any telephone. This Voice Web infrastructure gives callers instant access to personalized information, custom-tailored shopping, fully informed customer service agents, and can even send alerts when important events occur.

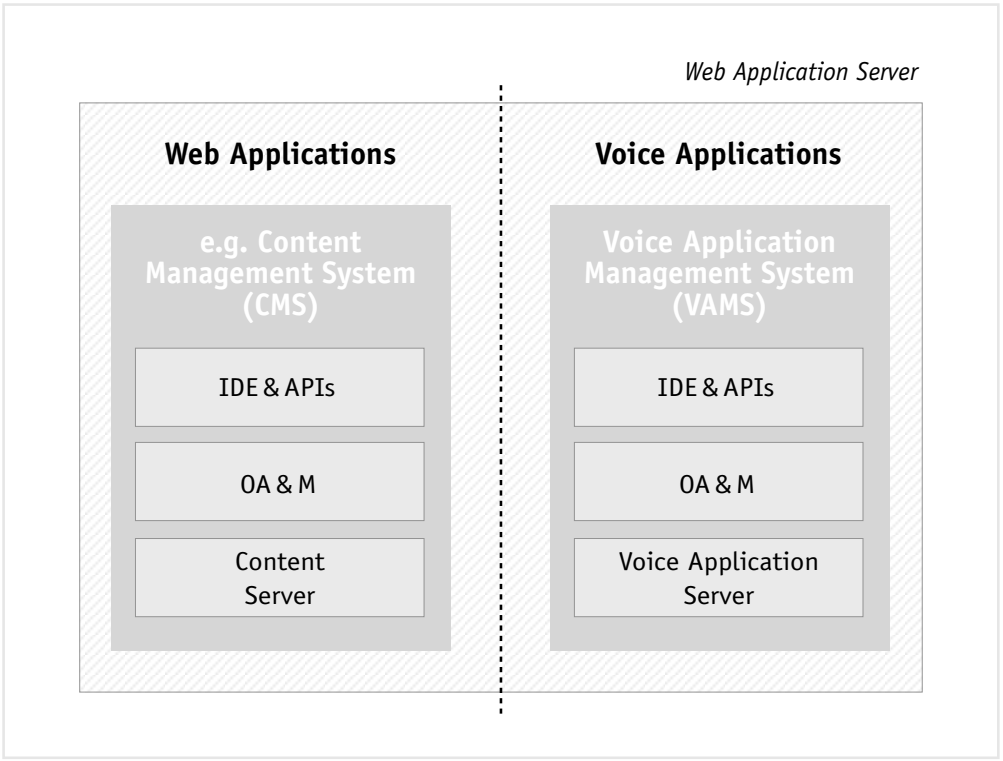


Figure 2:

Technology evolution for voice applications leads to Voice Application Servers

The two part technology concept of VoiceObjects X5 enables seamless and platform independent voice application development with a graphical Integrated Development Environment (IDE), and the final deployment of such voice applications on a Voice Application Server in the network. It uses widely deployed Web Application Server infrastructure and can easily be integrated with existing databases, contact centers, and content from Web sites. The VoiceObjects solution allows businesses to implement voice applications as easily as Web applications – using tools adjusted to the special requirements of a voice interface.

VoiceObjects X5 can be made available as an on-premise software platform, allowing customers to maintain full control over their applications. Or it can live inside the service provider network, providing a consistent level of service and interactions to many different companies. Enterprises, carriers, and service providers enjoy rapid application development, deployment, and administrative simplicity. The platform provides integrated maintenance with the advantage of full control of the voice user interface, dialog logic and the content that shapes end user experiences.

VoiceObjects' software platform is built on proven technologies and integrates with industry leaders such as Alcatel, Comverse, IBM, Genesys, HP, Nuance, Telisma, VoiceGenie, and others offering VoiceXML or SALT access for broad call handling functionality by themselves or through partners. The technology flexibility is derived from the VoiceObjects® principle that employs a construction kit approach for voice application creation and adaptable Speech Platform Drivers for the various VoiceXML or SALT dialects on the market.

Life Cycle of Voice Applications

Just like any other software applications, voice applications have a unique, yet definitive life cycle. They follow the same pattern of development – they are born as a concept, validated as a plan, deployed as an application, and after a set of maintenance and enhancement iterations ultimately retired.

Many voice application initiatives suffer from the "Just get it done!" philosophy of implementing the first project. Once deployed, the issues surrounding the ongoing maintenance of larger voice applications and voice portals presents what is most often an unexpected, underappreciated and unnecessarily high cost associated with voice application ownership.

Before standardization such as VoiceXML took place, voice applications were specialized projects that could only feasibly be run on the proprietary hardware and APIs of one vendor. The associated costs were typically high. But even with newer architectures leveraging VoiceXML, design and maintenance remains a cumbersome process of creating and assembling the components of a voice application in a manual fashion. Specialized tools are needed to handle the specific requirements of a voice-based information channel compared to other interfaces such as Web pages.

Moreover, life cycle management is a fundamental issue for managing any IT resource and will likely be a prominent aspect of the whole project. Yet to date it has been absent in most voice application deployments, because no specific systems for voice application management were available.

One approach to deliver such a system would be to take the familiar iterative model of other projects but adjust it to the specific needs of the voice channel. To simplify, we call the four stages of this interactive cycle: **Create, Test, Deploy** and **Analyze**. We envision the stages revolving around a Voice Application Server instance as the hub of all activities. Voice application life cycle management is a matter of multiple iterations of this cycle, illustrated in Figure 3.

While the process above is an idealized scenario of voice application deployment and maintenance, the ease with which it is realized is largely a function of the management systems chosen. The best option is an integrated voice application development and management system specifically designed to support iterative life cycle software management for voice.

VoiceObjects X5 is such a complete integrated voice application development environment, designed to automate the life cycle process of Create (design, development and integration), Test (functional, acceptance and system tests), Deploy (system integration, deployment and OA&M) and Analyze (data collection, reporting and CRM). Designing and operating voice applications for call center operators or mobile carriers, with access to corporate back-end data, is now an easier task to achieve.

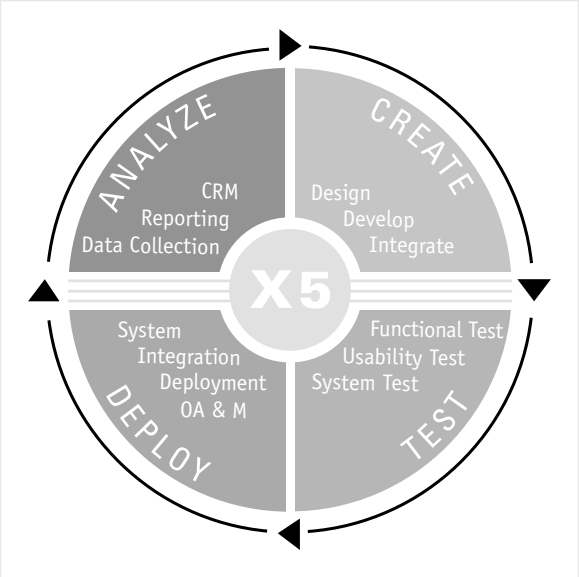


Figure 3:

Voice application life cycle centered around Voice Application Servers

Voice Application Fundamentals

The business case for speech-based applications is clear. Every day, companies across the globe are realizing the benefits. But how do we effectively Create and Deploy these applications in today's IT landscape? This section will focus on the fundamentals of modern voice application environments, starting with the prerequisite of an architecture that supports full life cycle management for voice applications.

Development is Internet Centric

Traditional speech services and DTMF applications were built on proprietary hardware and developed with vendor specific APIs. Today all of the server components in a modern voice application environment (e.g. VoiceXML Interpreter, Voice Application Server, Web Application Server, Database Server, Recognition Engine, Text-to-Speech, etc.) are linked through an IP network as illustrated in Figure 4 and can also be geographically distributed. This means that communication between Voice Applications and a VoiceXML Interpreter are analogous to the desktop Web browser model. This design offers organizations control of voice application development while re-using the existing components of the network.

In a standard Web-based application, servers typically issue instructions to desktop browsers like Internet Explorer using HTML, the mark-up language for Web sites. The browser responds by creating a page on a computer display and establishing links to other pages on the Web. When the desktop user clicks on a link, the browser issues a request that results in another Web page being fetched and displayed. The fetched resources are typically graphic images (GIF, JPEG, etc.) or text. However, as Web sites moved away from static content to dynamic interaction, sites needed more than HTML. Highly interactive Web sites (e.g. for e-commerce, information portals, etc.) now use both HTML code and components such as server side scripts. These components, often written in Java or Perl, manage the interaction with the user, while HTML is used to present the results to the user's Web browser.

This analogy is useful in understanding the role of VoiceXML and how this technology is handled in today's environments. Requests and information exchange on state-of-the- art voice applications are handled in a similar fashion, but the operative Internet communications format for this is XML and either VoiceXML or SALT.

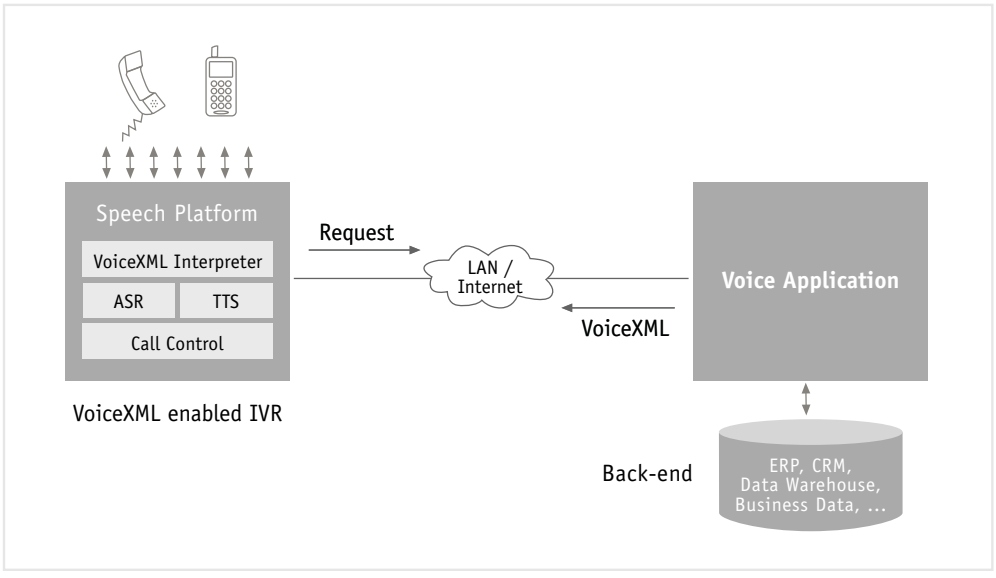


Figure 4:

Present day architecture for voice application deployment

These mark-up languages provide standardization for design and representation of voice dialogs. But developing complex dialogs with functions such as personalization and consistent dialog navigation requires reusable components and dynamic generation of such dialog components. Such personalization and dynamic creation can be achieved in two ways – either by using individual coding of server libraries or using a managed systems approach that generates dialogs out of proven components. Which is to say, either the hard way or the easy way!

Requirements for Voice Application Development

In order to provide a good and clean voice application, designers and developers need to create and maintain voice dialogs intuitively, quickly, and flexibly using reusable components. Designers should not be forced to maintain applications and associated application logic in file systems or other means that need a manual integration process. Dialogs should also be accessible from a central point of view where all resources are maintained and deployed in a consistent manner.

This section describes some of the key components of a voice application architecture comparing the two current methods for development and deployment:

- **Hand-crafted:** Hand-crafted creation and maintenance of voice applications and surrounding logic uses a server library approach. This approach also typically includes the use of front-end tools for design of application flow and logic, but these tools do not treat all necessary components as a single project, leaving the burden of integration, usability, and consistency solely to the developer.

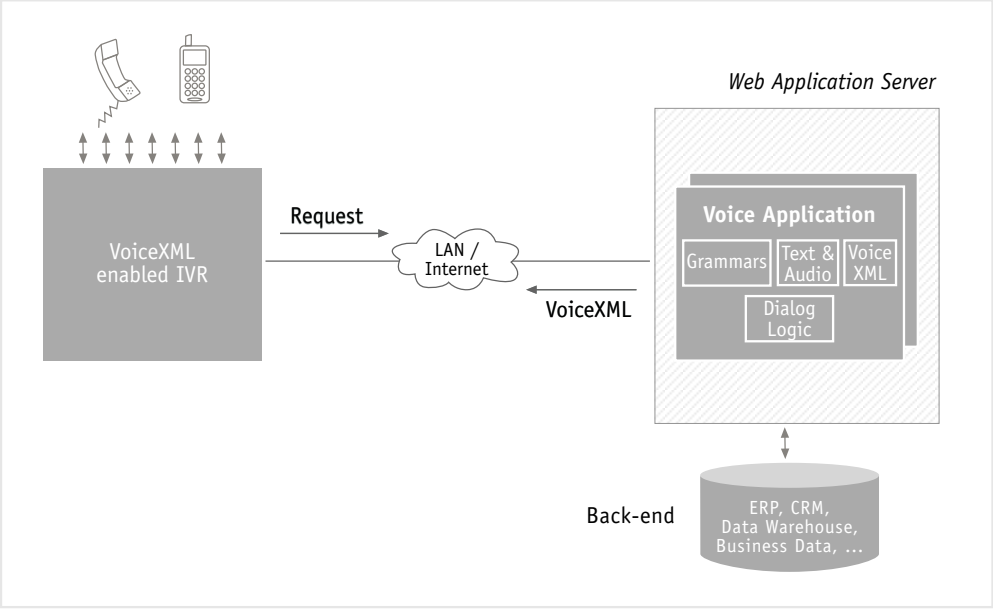


Figure 5:

Voice application development using hand-crafted coding

- **Process-driven:** Process-driven approaches create, maintain, and dynamically generate voice applications with Voice Application Servers as an essential component. Leveraging existing IT infrastructure, such servers make voice applications available in a familiar environment as an extension of existing applications

As mentioned in the paragraph above, voice applications can be built on an internet centric architecture, using open standards such as VoiceXML or SALT. Figure 4 illustrated the principle architecture for voice application development and deployment using VoiceXML as implementation option. The VoiceXML interpreter on the left translates each coded VoiceXML page into the necessary speech recognition, text-to-speech, and telephony functions.

The voice application on the right has to manage the user interaction, functional processing, and back-end integration, and provide VoiceXML mark-up for execution. Developers using non-specialized tools to design voice dialogs will typically use Java and XML development tools and editors to create voice dialogs by coding all required components of a voice application. Figure 5 illustrates briefly the resource management process that needs to be handled to obtain a functional voice application using a manual coding approach.

Voice Application Servers as Central Hub

In general voice application development presents a new IT paradigm, requiring different implementation expertise not typically used to create window-based graphical applications. Beside a different development life cycle the required skill level for designers and developers is high, even with mark-up languages such as VoiceXML or SALT. Experts need to combine speech processing knowledge, audio processing, voice dialog design, Java and database coding experience. Finding such a combination of skills is very difficult and frequently expensive.

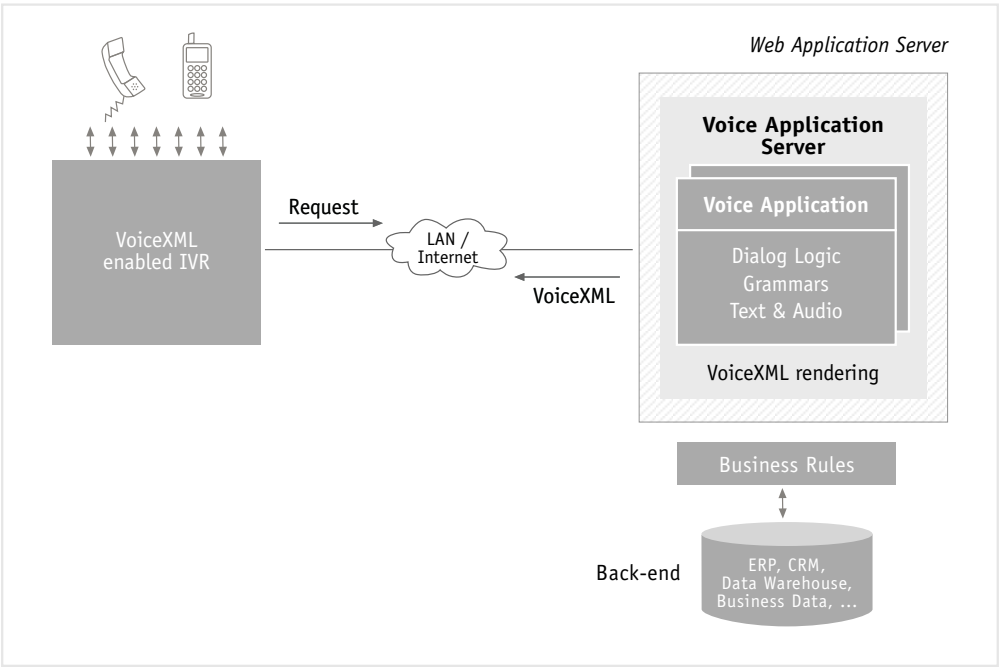
As a consequence of this, the voice application space would benefit from a powerful middleware platform architecture that helps automate and structure systems development while simultaneously using the existing infrastructure of Web application servers and Business Rules in these servers and adjacent enterprise back-end systems. This architecture would lower the skill requirements of developers and empower a much larger developer community. Figure 6 illustrates the role of such middleware architecture, employing a Voice Application Server as the central hub component that automates and structures the development of enterprise- and carrier-class voice applications. Since this does not assume the constructs and the associated learning curve of a full programming language, even non-programmers will be able to design and develop voice applications within such an architecture. The level of programming skill required by this approach is approximately the same as that needed to write an Excel spreadsheet.

The availability of standards-based technology and products for voice application development will fuel the rise of new voice services in the same way that industry standards enabled

the rapid growth of the World Wide Web. But Web sites are managed in coding environments that specifically address the needs of a Graphical User Interface (GUI). Something similar must be provided for the Voice User Interface (VUI) which is essentially a sequential presentation interface. Specific design rules and best practices apply that need to be addressed by a specialized Voice Application Management System (VAMS).

Figure 6:

Modern architecture using Voice Application Servers and associated tools



Dynamic Dialog Generation

What are the components needed for a complete Voice Application Management System that meets the special requirements of a voice interface within the **Create, Test, Deploy** and **Analyze** model?

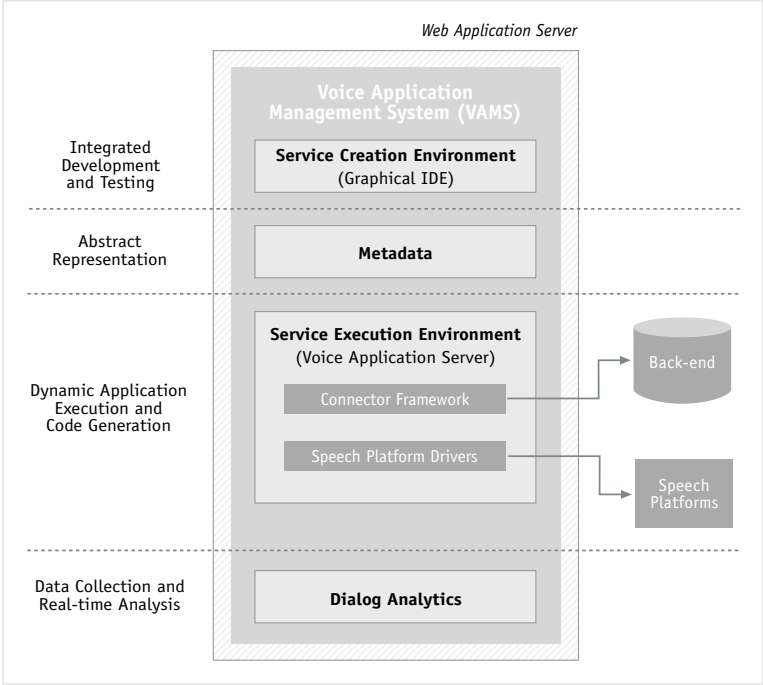
Taking the Voice Application Server as the corner stone of the overall architecture, as illustrated in Figure 7, the Service Creation Environment on the top provides the environment to support fast and easy development and testing of voice applications **(Create & Test)**. These applications will then be deployed to the Service Execution Environment (Voice Application Server) by basically assigning a telephone number to the applications **(Deploy)**. Users can call and use all branches of such an application instantly while also accessing actual enterprise data dynamically whenever it is incorporated into the dialog flow.

What characterizes an integrated **Create-Test-Deploy** process that allows creation of dynamic and personalized voice services?

- **Dynamic dialog mark-up generation** – Voice Application Servers can easily overcome the rough charm of older IVR systems that had silly dialogs like “Please press or say One”. Instead of frustrating callers we should be engaging them when they use a voice application. Dialogs need to attract callers with lively and flexible interactions resulting from personalized infor-

Figure 7:

Voice Application Management System for integrated voice application life cycle management



mation presentation. Good dialog can be compared to dynamic sites on the Web that generate a personalized view (e.g. MyYahoo, Amazon, etc.) to make a more interesting and relevant information presentation. As a consequence Voice Application Servers need to generate the dialog mark-up language (VoiceXML or SALT) dynamically at call time to facilitate personalization.

- **Access to back-end systems** – A voice application needs to incorporate existing back-end systems and data sources in an easy fashion. This permits lively and dynamic dialogs, and will reduce the maintenance costs of voice applications. No additional coding is required for dynamic menus or selection lists. This will overcome the burden of static dialog steps such as "Please select 1, 2, or 3" by allowing the user to say the name of the menu item directly. Nearly all applications will need to access existing data sources. In today's IT world nearly every business is database business. Voice applications can tap these information sources and reuse them for intelligent dialogs using standard access methods that are dynamically integrated into the generated voice mark-up code.
- **Reuse of existing business logic** – As mentioned above the most logical use of external systems is information presentation from existing databases or Content Management Systems. As these systems typically employ a Web interface and/or access structure, some business logic is usually implemented to obtain consistent behavior. Voice applications as an alternate channel to such information can make use of this infrastructure. Voice Application Servers are the foundation for building voice applications consistent with other communication interfaces such as a Web browser. For example, a Voice Application Server can leverage the same existing workflow rules for processing a service ticket within SAP R/3 or when buying a CD at Amazon.com.

Continuous Dialog Analysis & Maintenance

Once callers start using a voice application, a large volume of very useful information becomes available to the developers and business sponsors. Monitoring and logging capabilities of Voice Application Servers provide detailed information about dialog flow and usages that is essential for optimizing application deployments and for system maintenance. This data can also be fed back to integrated analytical procedures supporting dynamic dialog adaptation, that generates better personalized dialogs. In effect the **Analyze** step of the proposed life cycle model will now be available to both the developer and business experts of the company. This approach substantially strengthens the involvement of the business interests in the ongoing operation of the system.

Recently, click stream analysis has become quite important for Web designers. Web site visitors surf through web sites and their trace can be usefully analyzed in CRM projects to determine consumer segmentation and other analytics that better define future web offerings. Click stream information is collected on an object level rather than a simple page view for better analysis.

Since voice application environments can be based on Web infrastructure as well, similar concepts can be applied. Dialog stream analysis is a function of the Voice Application Server that controls and logs all interaction with a caller and governs all information for usage and dialog analysis. It's designed to provide detailed information on the caller interaction in order to optimize application deployments or personalize a dialog. VoiceObjects X5 provides a deployed data scheme and extraction and loading facilities, stored in relational databases for detailed analysis. Standard integrated OLAP (OnLine Analytical Processing) functionality allows aggregation, slicing and dicing of available dialog usage information. This information can be fed back to designers and business analysts to make intelligent design, deployment, and marketing decisions to continuously improve a voice application.

In order to provide dialog designers, marketing managers, and also system administrators with the data they need to evaluate voice application performance, and to pinpoint those areas that may require optimization, complete Voice Application Management Systems should provide reporting templates as a "jump start" solution.

Sample reports might cover the following:

- Administration Reports for System Management (System Administrator)
  - Performance Trend Analysis
  - System Throughput Monitoring
  - Server Load and Availability
  - Execution Cycle Breakdown
  - System Capacity Planning

- Application Maintenance Reports (Designer, Developer, Systems Integrator)
  - Recognition Trend Analysis
  - Event Correlation Analysis
  - Service Termination Breakdown
  - Navigation Usage
  - Slow Moving Dialogs
- Application Usage Reports for CRM (Product Manager, Sales & Marketing)
  - Airtime Analysis
  - Quality of Service
  - Service Stickiness
  - Hotspot Analysis
  - Service Correlation

Dynamic Personalized Voice Applications

Often overlooked in traditional voice application creation, personalization is a process that builds stronger loyalty and better information presentation for the caller. Web site designers already use implicit or explicit personalization to better address Web site offerings to consumers. The same can be achieved with an integrated Voice Application Management System that employs similar mechanisms of voice dialog personalization. Businesses benefit from converting customer preferences into tailored dialog and service offerings that help to increase voice application usage in time and/or frequency – and thereby drive higher revenue or minutes of use. From the moment a caller can be identified by a voice application, the goal is to create a personal relationship between the machine and the human caller. From the caller's perspective, that relationship is supported and reinforced most effectively by personalization of the dialog and the voice user interface. Customer activities throughout the dialog can increase the level of personalization in a cumulative fashion. After several visits, the voice application might even greet the customer as an "old friend" or as a professional user. Purchasing preferences, such as payment methods, shipping options, product categories and affinity interests can be retained from all visits (maybe even combined with Web visits of the caller) and used to improve customer personalization even further. Voice Application Servers can use existing IT back-end systems and standard connectivity to leverage existing enterprise systems and databases to obtain and store such information.

Important to efficiently manage the different application scenarios is an integrated layer concept. Layers allow designers to easily separate the core application logic from the way an application is presented. Layers can influence aspects of an application such as language or Persona, but they can also change the application behavior itself by applying customer profile information from a customer database after caller authentication. Designers may flexibly stack any number of such layers on top of each other when building their applications. The Voice Application Server performs the task of dynamically combining them at call time, rendering the correct dialog thus eliminating the need to manually consider all of the different combinations that may occur at development time.

Calling the Right Architecture

Any new technology requires time to understand. As we invest time in understanding it, many of the general concerns will disappear. This is true for voice applications and the associated new architecture based on open standards and Web-like installations. To put it simply, Voice Application Servers generally sit on top of Web Application Servers leveraging several components of the IT infrastructure. This includes already available business logic deployed on the Web Application Server side. Integration with speech processing and telephony resources is achieved using standard Internet connectivity (http connections). Figure 8 illustrates the components of such an architecture.

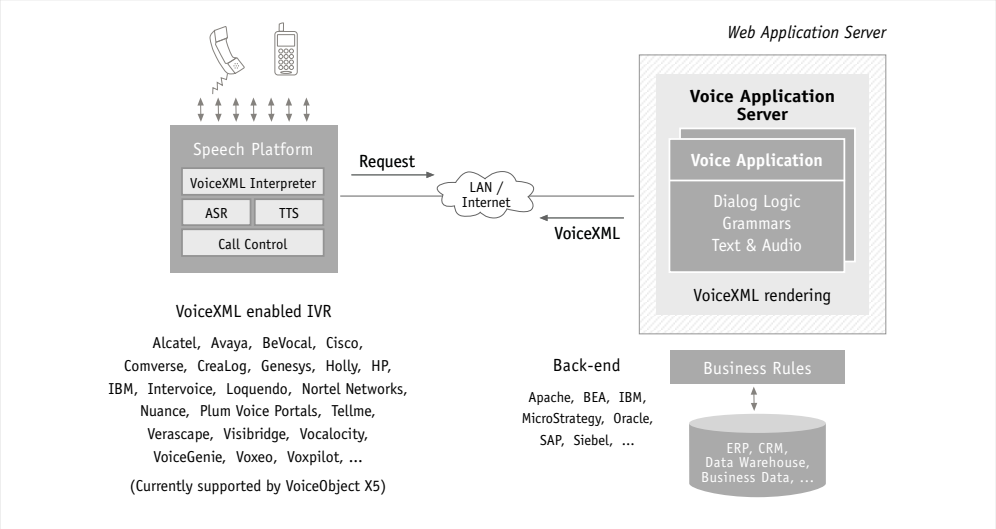


Figure 8:

Voice Application Servers leverage existing IT infrastructure

The Voice Application Server controls all interaction with the caller. It includes the management of all grammars that specify the words and phrases that can be recognized, the associated actions for each spoken command, and the audio resources. According to the application logic it then generates the necessary voice mark-up statements (e.g. VoiceXML) for execution.

But like other IT systems, a high volume voice system needs to fulfill certain criteria with regard to standards, availability, scalability and maintainability.

Voice Application Servers need to Support Standards

Traditional voice services and DTMF applications on proprietary hardware are outdated and unnecessarily complex for creating and maintaining new services. Voice Application Servers eliminate the proprietary coding components and provide an infrastructure that is flexible and expandable across voice platforms. Standards like VoiceXML are now supported throughout the industry. Businesses support VoiceXML because it provides many compelling benefits. As a mark-up language, VoiceXML allows companies to interact with almost all of their existing Web applications and it drives voice applications into a new era of economics and vendor independence.

VoiceObjects takes this concept to the next level by implementing speech platform drivers. These drivers allow VoiceObjects to be agnostic with respect to different VoiceXML and SALT language flavors or dialects. Using VoiceObjects, voice applications can run on any of these platforms regardless of the dialect. This makes the VoiceObjects platform ideal for flexible voice application development and deployment. It protects your application investment no matter what physical or software platforms evolve in the future.

Voice Application Servers need to Scale

Scalability for voice applications means that the more callers call an application the more lines must be supported. In general more resources in terms of telephone line, ASR, TTS and VoiceXML gateway capacity must be added to accomplish this but also Voice Application Servers need to provide more and more "lines" to generate dialogs for active callers. Voice Application Servers in particular must ensure the following:

- **No bottlenecks to scaling** – New servers can be easily and dynamically added to the configuration as necessary to meet increasing caller demand, and that the overall request load is optimally distributed among the servers so that resources remain fully utilized.
- **No single points of failure that could impact availability** – Requests must automatically failover from non-working components to working components. Moreover, application and user state must be protected to ensure that any failures that occur (e.g. server crash) can be fully masked from the user and the application.
- **Single-system image to your administrators** – System administrators must be empowered to manage voice services and servers as a single logical resource. This greatly simplifies operations and helps prevent any inadvertent introduction of inconsistencies.
- **Transparency of the mark-up platform decision (VoiceXML or SALT)** – Employing a driver concept for different speech platforms application designers should not have to deal with intricacies of platform differences. By not relying on specific platform features but more on a higher level component based architecture, application investment is protected as organizations move voice applications from one platform to another or even deploy
- **Hardware and operating system independence** – Using a Java-based software environment as a foundation, Voice Application Servers can be established across disparate hardware, operating system, and web application server platforms.
- **Multi-tenant capabilities** – While systems grow the Voice Application Server must support to operate multiple tenants on a single system including completely separated logical units.

  For many of these requirements Voice Application Servers could leverage functionality from Web application servers but they definitely also need to store, retrieve, and maintain dialog specific context information from own resources supporting installations of thousands of lines and scale to support millions of parallel dialogs.

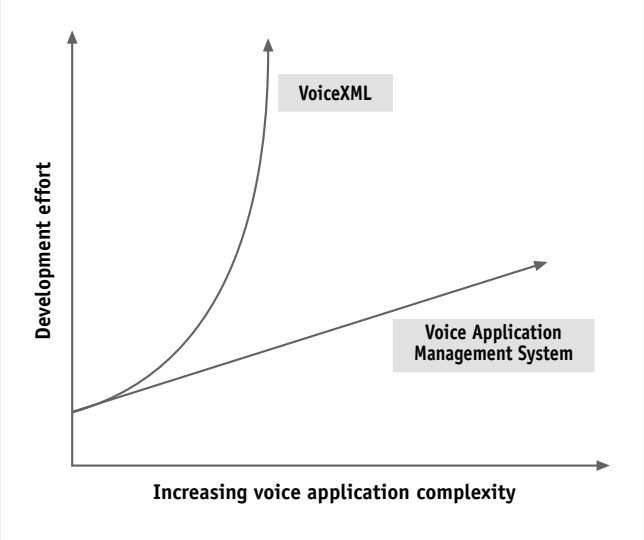
Developers need to Scale too

Sophisticated dialog requirements result in complex voice applications that must be managed intelligently. While there is no straightforward definition, there is a spectrum of sophistication, from simple dialog at one extreme (e.g. mainly gathering a single piece of data that drives some action) to extremely sophisticated conversational dialogs with dynamic data integration, personalization, and a high degree of usability for the caller (e.g. collecting a set of information, verifying, confirming and correcting them easily) at the other end.

Providing the required support for such dialogs in pure VoiceXML coding will increase the complexity of the project, thus potentially placing it out of reach of the target developer audience to create and manage such a project. In some instances VoiceXML and SALT provide the basic capability, but do not provide default implementations or prebuilt templates. Therefore developers need to assemble their own custom libraries of templates or modules and will also need to maintain them. The net result is that simple tasks are easy to do in VoiceXML, but sophisticated voice applications become increasingly challenging, requiring a non-linear increase in effort to manage such an application.

Figure 9:

Sophisticated voice dialogs require non-linear increase in effort when coding in traditional ways



A novice developer might choose to first write test applications in pure VoiceXML as long as the dialogs are relatively simple and without back-end integration, e.g. playing simple prompts. But these developers will find it challenging to extend their initial work to complex voice applications. Fortunately a complete Voice Application Management System would allow them to graphically develop new applications and deploy them on a Voice Application Server using a simple “one click” assignment. Developing and deploying an increasingly complex voice application without a specialized management system inevitably results in a growing chasm between the time to build a new voice application and the degree of technical sophistication required to build and manage it.

Within large projects typically multiple users work on different tasks and have different roles, requiring a system environment for collaborative development and integrated user management.

VoiceObjects X5 provides centralized User Management as part of a platform-wide Metadata Repository to maximize multi-user collaboration. All users have roles, which gives them dedicated privileges according to their skills and respective project tasks. The provided user roles and privileges are aligned along tasks for design, development, controlling & monitoring, and system administration. Audit trails offer tracking of individual user activity.

Conclusion

The ability to act quickly and with quality solutions in today's increasingly competitive marketplace is critical to the success of organizations. To meet these business requirements, voice applications need to 1) be developed and deployed in a short time, 2) scale to large numbers of callers, and 3) leverage and integrate enterprise back-end systems to generate a personalized dialog. We find that of the two main development approaches, only one can meet these needs:

- **Integrated Voice Application Management Systems** allow voice-specific design, development, and maintenance, as well as an easy integration of external back-end systems. Voice Application Servers as a general architecture serve as a hub that scales to meet a wide variety of voice applications ranging from deployments at enterprise call centers up to carrier grade Voice Portals.
- **Hand-crafted server libraries** are a solution that is suitable for first trial or departmental systems with limited voice application complexity and lower application scalability in terms of the number of callers. Limited re-usability and reduced dialog convenience will become an important issue during the manual life cycle process for such services.

As we look at the voice application development area, we believe that VoiceObjects made a pivotal decision in creating a Voice Application Server. The server has positioned the company well ahead of others still offering the manual programming approach for voice applications and at least 12-18 months ahead of its nearest competitor in complete Voice Application Management Systems technology.

The VoiceObjects Advantage

With a VoiceObjects powered solution, enterprises, carriers, and solution providers can gain the advantages of creating their own voice application environment based on standards such as VoiceXML or SALT but without being bound to one specific IVR platform vendor. Organizations are released from the burden of buying a complex telecommunications infrastructure with a fixed set of applications but instead can integrate modern voice solutions into their normal IT and telecommunications infrastructure with full control over their applications.

Within a VoiceObjects environment, the previously independent worlds of voice and data applications can be merged into one. VoiceObjects' ability to create and add sophisticated voice dialogs to Web-based applications allows the voice and data communities to share business rules, programming logic, customer data, back-end integration, and customer contact management systems. Moreover using a specialized set of tools, unique to the voice channel, VoiceObjects allows a reduction in skill level for creating useful telephony-based voice applications.

Use of normal Web infrastructure such as Web Application Servers allows organizations more efficient development, leveraging existing installed systems and deploying new services more rapidly. Sophisticated customer relationship management applications already in place can be used to provide a highly personalized caller experience by dynamic generation of voice dialogs (similar to dynamic web pages like MyYahoo, etc.). Integration with call routing solutions at call centers or telephony switchboards allows voice applications to behave as an extension of the normal telephony system. Intelligent voice applications enhance productivity and reduce costs.

A VoiceObjects powered voice application environment provides access to highly sophisticated technology under the control of the normal IT infrastructure. In this environment, businesses can afford the luxury of focusing their time and talents on creating a superior, scalable, personalized customer contact experience.



VoiceObjects

all business is vocal

Glossary

API Application Programming Interface – An interface that applications use to access other services and to communicate with other systems.

ASR Automated Speech Recognition – Modern systems that use speech instead of DTMF as input and respond to the recognized utterances.

DBMS Database Management System

eCRM Electronic Customer Relationship Management

HTML Hypertext Mark-up Language – The most popular mark-up language in use today. HTML is one of the foundations of the web development, providing formatting and basic structures to documents for presentation via browser applications.

IDE Integrated Development Environment – A programming environment with a set of tools to support the design, development, deployment, analysis, and maintenance of an application.

IVR Interactive Voice Response – Systems that traditionally hosted DTMF-based applications. They prompt the inbound caller for information using a recorded or synthesized human voice. Current systems work with ASR and TTS functionality to allow a better VUI.

Java An object-oriented programming language developed by Sun Microsystems. A write once, run anywhere programming language often used in today's Web application development.

Metadata In voice application terms it provides a means to query and match components of dialog definitions. This allows systems such as VoiceObjects to retrieve components, based on the metadata that describes the components, and then to generate an active voice dialog out of the retrieved components. See also VoiceObjects below.

OA&M Operation, Administration and Maintenance - Tools and functions to help execute administrative task to operate and deploy systems and services.

OLAP OnLine Analytical Processing – Decision support principle that allows the user to quickly analyze information in multidimensional views and hierarchies. In voice terms for example, OLAP tools can be used to perform trend analysis on dialogs and machine load for call processing. This will enable developers to drill down into masses of dialog statistics in order to isolate problems with a voice application.

SALT Speech Application Language Tags – A mark-up language intented to support multimodal interactions. SALT was created by the SALT forum (www.saltforum.org) with a mission to develop standards for speech enabling HTML and XHTML. SALT also supports a language subset for "voice only" specifications. The SALT forum contributed SALT to the W3C for their work on multimodal interactions.

TTS Text-to-Speech – Describes a system that uses synthesized speech to provide information or enable a dialog with the user. Usually a specific TTS engine is needed for each different language.

VAMS Voice Application Management System – A system that manages a collection of dialog objects (VoiceXML code, audio files, grammar files, and so forth), including metadata about the content. Voice application management systems like those marketed by VoiceObjects usually provide services such as retrieval of dialog objects, including creation, editing, and versioning in a graphical manner. Also analytical capabilities complement the life cycle process for voice application development to allow designers fast adjustment/enhancement of active voice applications.

v-commerce Basically describes voice driven e-commerce over the phone. V-commerce leverages the same application servers, business rules, and databases from existing systems to serve both phone and desktop users.

VoiceObjects Term used by VoiceObjects to describe its dialog components which completely define a voice application (e.g. objects for menus, forms, reports, filter, database access, etc.). Aside from standard dialog components the library includes definitions for logic, resources like grammars and audio files, and actions. VoiceObjects® are stored as metadata in a central data repository in a relational DBMS.

Voice Portal Voice Portals are similar to Web Portals, offerings a broader variety of voice applications to the caller using just one telephone number. Initial voice portals are largely horizontal – offering synergistic information services and base commerce applications centered on entertainment, convenience, and personal productivity.

VoiceWeb Is the term for an emerging network of voice applications and Voice Portals that people can access from any phone to obtain information or connect to other phone numbers in one, continuous session. Similar to the Web, the VoiceWeb will not happen instantly but evolve over time.

VoiceXML The de-facto standard in mark-up languages for creating voice applications. The VoiceXML forum (www.voicexml.org) as the inventor of VoiceXML is an industry consortium dedicated to promoting VoiceXML as the standard for phone and voice retrieval. The current VoiceXML 2.0 was expanded to support mark-up languages for grammar, speech synthesis, and call control. Multimodal specifications will further expand the standard. VoiceXML is now maintained by the W3C.

VSP Voice Service Provider – A special form of an ISP (Internet Service Provider), a company that hosts a voice infrastructure for other companies. VSPs play an increasingly important role in hosting voice applications that lets companies and network carriers bring voice services or Voice Portals to market quickly.

W3C World Wide Web Consortium – Internet standards body responsible for many of the standards key to the functionality of the world wide web today. The W3C is working on voice expansions to access Web content. See www.w3c.org/voice for details.

Web Application Server Software that enables Web applications to exchange data with the back-end systems and database of e-businesses. Web application servers include Apache Tomcat, Bea WebLogic, IBM WebSphere, and others.

XML Extensible Mark-up Language (XML) – The universal format for structured documents and data on the Web, XML is increasingly becoming the general standard document format of structured data. For more information, go to www.w3c.org/XML.

This document is an indicative draft only and is accordingly subject to amendment and verification and may contain bugs or inaccuracies that are latent or otherwise not discovered. VoiceObjects AG or any of its subsidiaries expresses no implicit or explicit warranties of any kind to the accuracy or suitability of this documentation nor does VoiceObjects AG or any of its subsidiaries accepts any liability in respect of its use by any party for any purpose.

VoiceObjects is a registered trademark of VoiceObjects AG in Germany or other countries. VoiceObjects X5, VoiceObjects Server, VoiceObjects Desktop, VoiceObjects Infostore, VoiceObjects Analyzer, VoiceObjects XDK, VoiceObjects Studio, VoiceModules, VoiceTemplates, and "all business is vocal" are trademarks of VoiceObjects AG in Germany or other countries. Any other trademarks, trade names or service marks mentioned in this document belong to their respective owners.

The material presented herein is based upon information that we consider reliable, but we do not represent that it is error-free and complete. VoiceObjects is not making any representation or granting any warranty with respect to such material, and the distribution of such material shall not subject VoiceObjects to any liability.

Copyright 2001-2005 VoiceObjects AG and its licensors. All rights reserved.

VoiceObjects AG, Friedrich-Ebert-Str., 51429 Bergisch Gladbach, Germany

Published in Germany – Legal information July 2005

CENWP_X5_VAMS 2005-07



VoiceObjects

all business is vocal

Headquarters EMEA: VoiceObjects AG, Friedrich-Ebert-Strasse, 51429 Bergisch Gladbach, Phone: +49 (2204) 845-100, Fax: +49 (2204) 845-101

Headquarters Americas: VoiceObjects Inc., Burlington Centre, 35 Corporate Drive, Suite 190, Burlington MA 01803, Phone: +1 (781) 761-1500, Fax: +1 (781) 761-1501

Office UK: VoiceObjects Ltd., 1 Northumberland Avenue, Trafalgar Square, London WC2N 5BW, Phone: +44 (870) 351-6748, Fax: +44 (870) 351-6749

www.VoiceObjects.com