The Voice Web:
A Strategic Analysis

by

David E. Pearah

B.S. Computer Engineering
University of Illinois at Urbana-Champaign, 1996

Submitted to Engineering Systems Division and the
Department of Electrical Engineering and Computer Science
In Partial Fulfillment of the Requirements for the Degrees of

Master of Science in Technology and Policy

and

Master of Science in Electrical Engineering and Computer Science

at the
Massachusetts Institute of Technology
February 2002

Signature of Author _____
Department of Electrical Engineering and Computer Science
September 7, 2001

Certified by _____
Lee McKnight
Associate Professor of International Communication
Thesis Advisor

Accepted by _____
Arthur C. Smith
Chairman, Committee on Graduate Students
Department of Electrical Engineering and Computer Science

Accepted by _____
Daniel Hastings
Director of Technology and Policy Program
Engineering Systems Division

DEDICATION

"Sometimes a scream is better than a thesis."

- Ralph Waldo Emerson (1803-1882)

In the course of preparing this discourse on the Voice Web market, I have become fully aware of my indebtedness to those who have helped me throughout my tenure at MIT and in the communications industry:

- Dr. Lee McKnight, for his abiding guidance and leadership in the Research Program on Communications Policy and the Internet Telephony & Convergence Consortium. He gave me the freedom to pursue the topics that interested me (which were many and sundry), and mentored me even after his departure from MIT. Other key individuals in this group include Sharon Gillett (whose work on HFC and ISDN networks is still the golden standard for graduate research) and Dr. David Clark (a technological pioneer in the Internet community who is deeply involved in issues of pricing and universal access).

- Anthony Flynn, for giving me the opportunity to blossom in my role in product management and business development at PhoneRun.

- Since I do not fancy myself a writer, I am indebted to the editing services of Lisa Goren, former teammate and friend from PhoneRun.

- My various colleagues in the close-knit speech community, perhaps the most insular and yet visionary community I have ever participated in. I dedicate this paper to our shared success in popularizing this important technology.

TABLE OF CONTENTS

TABLE OF FIGURES

Chapter 1. INTRODUCTION

> "The term *Voice Web* should not be understood as simply extending the Internet to conventional telephones... Telephone voice interactions are fundamentally different than visually-oriented Web browser interactions... *Voice Web* is a good term to summarize a major opportunity, but the analogy to the World Wide Web should not be used to mask the critical differences."[1]

If the emergence of the World Wide Web was the communications and business phenomenon of the 1990's, then the Mobile Web is already poised to be the darling of the next phase of the personal communications evolution. Consumers are already bombarded with a panoply of mobile communications products and services: wireless telephony, cellular handset mini-browsers, short message services, text paging. Driven by either personal or business habits forged on the traditional Internet, the perception is that consumers are increasingly turning to technologies that enable continuous or ad hoc information access in mobile situations, such as driving in a car or walking down the street. These technologies are primarily viewed as complements to, and not replacements for, the traditional modes of information access.

As the name would suggest, the addition of the term "Voice Web" to the lexicon suggests yet another mode of mobile information access: speech interaction. While many of the technologies that support general speech recognition applications are hardly new, the consumer is infrequently exposed to anything more substantial than the most trivial of interactions, e.g. "Press or say '1.'" The relative anonymity of speech-interactive applications, coupled with the intoxicating (if not gratuitous) use of the term "Web," has created the perception that a new era of information access is about to dawn on the mobile consumer landscape, e.g. The Kelsey Group forecasts that by the year 2005, 45 million users will propel the voice market to $12 billion annually, more than $5 billion of which will be generated by advertising and commerce.[2]

The concern is not whether such estimations or projections are merited, but that the uncritical acceptance of the patently ambiguous Voice Web overlooks the multiplicity of dynamic forces that will come to bear on the evolution of the voice market in general.

---

[1] *The Voice Web and the Telephone VUI*, Editor's Notes (May 2000), Dr. William S. Meisel. See http://www.tmaa.com/voice_web.htm.

[2] Kelsey Group Press Release, "The Kelsey Group Forecasts $12 Billion Voice E-cosystem in 2005," 10 March 2000. See http://www.kelseygroup.com/pr000310.htm.

- How does the structure of the historically vertically-integrated speech industry affect the evolution of the market?
- How will applications be accessed, e.g. democratic web of sites or carrier-specific private network?
- Are all the necessary technology platforms, networks, and standards in place to support the desired vision of the voice market?
- Who owns the direct relationship with the consumer, and how does service pricing affect both the quantity and quality of applications?
- What are the trademark issues involved in providing access to audio content already on the Internet, i.e. does a voice service provider need CNN's permission to play audio files freely accessible on the CNN website?
- What limitations does the telephony voice user interface impose on the application space?
- Ultimately, will the applications being proposed provide significant value for the consumer, thereby creating sufficient demand to seed further development?

Given the complex interplay of the technological, economic, regulatory, and even human factors suggested above, the realization of the Voice Web concept is certainly not assured. As the opening quote from Dr. Meisel suggests, the pre-existence of the speech and Internet industries should not lull the reader into assuming that these two worlds will naturally merge into a single coherent and intuitive service model. While recent standardization efforts indicate a desire to extend the Internet service framework to voice applications, there has been little critical analysis of whether these technological innovations are sufficient to the task; moreover, other market factors (e.g. economics) are accorded little consideration in light of their potential impact on the evolution of the industry. For firms individually and the industry collectively, the success of the Voice Web ultimately depends on the anticipation of and appreciation for these interdisciplinary factors. This paper represents an attempt at a framework for both enumerating and negotiating these complex interrelationships, for the purpose of answering the following question: "**Will a Voice Web model naturally arise from existing industry structures and market forces?**" or conversely, "**What is required to ensure the realization of the Voice Web?**"

1.1. Scope of Analysis

Before outlining the methodology used to answer these questions, the scope of the analysis must be defined. Given the Voice Web's provenance in the previously disparate speech and Internet industries,

one of the major challenges of this paper has been to define a framework sufficiently broad to incorporate all of the relevant market factors yet suitably narrow to ensure practicality. For example, the incorporation of speech-interactive technologies on personal digital assistants (PDAs) portends consequences for the evolution of the traditionally telephone-centric speech industry, yet the reader would not profit from an abstract discussion of the entire PDA industry. Excessive contemplation of the nature of the Voice Web leads to the accurate but useless observation that it is but one facet of the entire communications industry, so the lines must be drawn, albeit arbitrarily, for the sake of clarity.

1.1.1. Terminology

Part of the difficulty in setting the limits of this discussion stems from the imprecision of the vocabulary itself, which has been liberally borrowed from the argot of the speech and Internet industries. For example, what does the term *Voice Web* mean?

- **Speech-interactive access to existing World Wide Web content?** There are already software packages that assist mobility- and vision-impaired individuals in accessing the Internet, as well as popular dictation programs that allow web browser control with simple voice commands.[3]

- **A single network through which multiple speech-interactive applications can be accessed?** Again, this already exists vis-à-vis the telephone network For example, a consumer can trade stocks through Charles Schwab at 800-435-4000, make a United Airlines reservation at 800-824-6200, and purchase move tickets through MovieFone at 617-333-FILM. While these telephone numbers more closely resemble numeric Internet addresses than the mnemonic domain names to which web surfers have become accustomed, this does not detract from the fundamental observation that a variety of speech-interactive information and entertainment services are available through the existing telephone network.

- **A network that allows voice navigation to multiple speech-interactive applications from a single phone number?** A variety of new and existing firms provide access to multiple content sources from a single number, including Lycos, Qwest Wireless, Tellme, and BeVocal. Yet to

[3] The Conversay Web system supports voice-activated Web browsing and navigation which is "great for people who have difficulty using their hands." (http://www.conversay.com/Solutions/Web.asp). The latest release of IBM's popular ViaVoice desktop dictation software allows voice navigation of the Web (http://www-4.ibm.com/software/speech/desktop/).

bestow upon each of these services the moniker of "Voice Web" would be akin to assigning the label of "World Wide Web" to a closed, proprietary service like America Online. Whether visual or vocal in orientation, sites that offer links to various types of applications and organizations are simply "portals," not the web itself.

- **A parallel publishing network distinct from the existing World Wide Web but reachable over the existing Internet?**[4] This vague definition only begs the question. For example, this would seem to disqualify voice applications that use the telephone network instead of a direct data connection, which represents practically every single speech-interactive consumer service ever conceived. Perhaps the definition qualifies applications that use any data or logic residing on the Internet, but this is clearly too inclusive since only the most trivial applications fail to use Internet-accessible resources, e.g. Charles Schwab's trading service relies on market data from both internal (account holdings) and external (quotations) sources.

Clearly, none of the definitions above adequately define the scope and leave many concerns unresolved. If the Voice Web interface is standardized around the telephone, other speech-capable devices (e.g. computers and personal digital assistants) are unnecessarily excluded from consideration. If the Voice Web applies only to mobile information access, then desktop computers and even wireline telephones are barred from participating.

In defining the Voice Web, the approach taken by the author is to adapt and leverage those characteristics that have facilitated the success of the "Traditional Web":

1. **Only one Voice Web**. Much of the Web's success derives from the inclusiveness and ubiquity of the publication process: once a website is on the Internet, it is accessible from any location on the Internet. The Voice Web should exhibit similar efficiencies in both scale and scope, stimulating an entire market to service all facets of content publishing, including design, implementation, hosting, distribution, etc. Not only the technology but also the economic incentives must be aligned to render seamless the connection between end users and publishers.

[4] Most consumers use the terms 'Internet' and 'Web' interchangeably since their experiences are typically limited to navigating between various sites (including those sites that provide email service), but the former is a collection of documents whereas the latter is a data network. More specifically, when 'Internet' is spelled with a capital 'I,' what is meant is the global assemblage of internet protocol-based networks, both privately and publicly owned, that have implicitly or explicitly agreed to interconnect for the purpose of forming a single common carriage inter-network.

2. **Standardization of protocol and document formats.** In support of the need for a single Voice Web, there must be considerable agreement on the syntax of relevant application or dialog languages, as well as the communications protocols through which they are conveyed. In the case of the Traditional Web, the Hypertext Markup Language (HTML) is the document standard, and the Hypertext Transfer Protocol (HTTP) builds on the existing Internet protocols to form the communications standard. Recent initiatives to coalesce the voice industry around several key speech-interactive application standards lay the foundation for further Voice Web development efforts.

3. **Fair, consistent, and intuitive access to all Voice Web sites.**[5] A corollary to the "One Voice Web" criterion is that the connection between consumers and destination sites cannot be intentionally compromised. While the Internet is comprised of many private entities that reasonably seek to maximize profits, the economic and technological configuration of the communications industry adamantly enforces the right of any consumer to reach any Internet-connected host. In addition to this technological model of "democratic" connectivity, the task of accessing and recalling HTML site locations is made easier by the introduction of fair, universal, and mnemonic naming services (e.g. consumer.net is much easier to remember than the equivalent numeric address 206.156.18.122), so one might expect that a similar system be indispensable to the Voice Web as well.

4. **Significant reuse of HTML Web[6] resources and practices.** To the extent that the Voice Web is complementary to the HTML Web, its creators would do well to ensure that this nascent industry leverage as many elements of the HTML publication process as possible: web servers, networks, document and protocol standards, best-practices code generation and management, even personnel. This does not imply that that the existing HTML infrastructure is suitable for supporting the Voice Web architecture, but it does suggest a natural starting place from which to expand and evolve.

This definition of a Voice Web is admittedly arbitrary but it serves to thoroughly distinguish this type of speech-interactive application network from other architectures. There is no presumption that the Voice Web will be accessed solely by mobile users, nor is there any bias against next-generation access devices

[5] Note that this criterion requires neither consistent nor intuitive access within each site, since one must always allow for poor site design and implementation.
[6] Though the expressions 'HTML Web' and 'Traditional Web' are synonymous, the former is preferred to the latter because the word 'traditional' is often associated with 'old-fashioned' or 'outmoded.'

(including desktop and palmtop computers). The Voice Web will certainly avail itself of certain resources and practices already employed on the Internet, but such reuse must be tempered with a critical eye towards exposing and overcoming fundamental mismatches. While speech-interaction is the hallmark of the Voice Web, there is no embedded restriction against other modes of input or output interaction; indeed, the Voice Web may be viewed simply as a recent addition to the greater World Wide Web (a.k.a. Web). The overlap between the Visual and Voice Web in Figure **Error! Not a valid link.** below suggests that the inclusion of these various modalities within the Voice Web model may result in an entirely new concept of the Mobile Web.

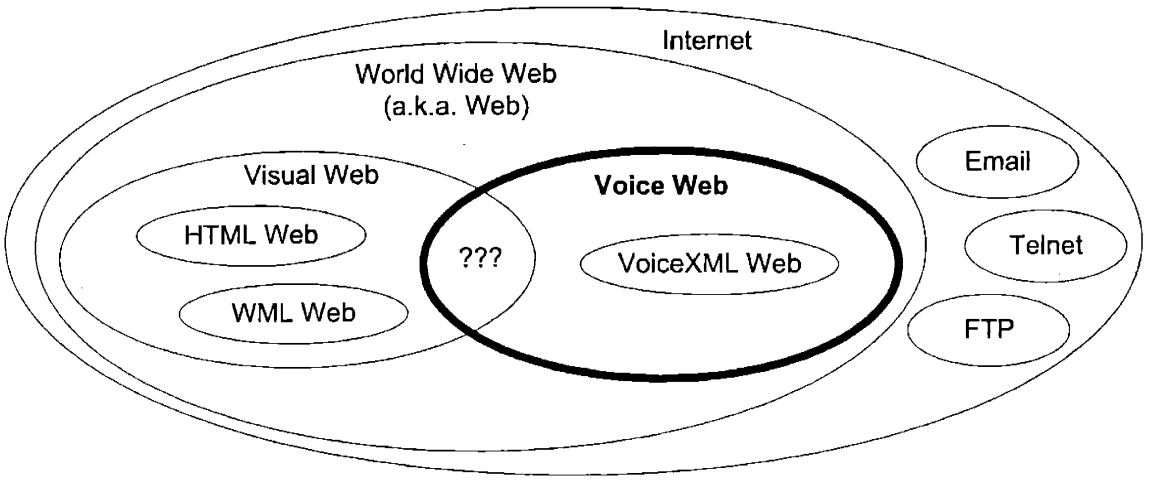

Figure 1. Voice Web's Relationship to the Rest of Internet

In defining the scope of the Voice Web, what falls outside of the definition may very well be the more interesting aspect, the most notable example of which is the Voice Portal. Except for perhaps final criterion listed above, Voice Portals fall well short of the essential characteristics of the Voice Web. Instead of a single connected network, each portal is a "walled garden" unto itself. Despite recent standardization efforts, there is significant deviation among the various document standards and programming conventions supported by each portal. Perhaps the most glaring omission is an objective system for assigning and resolving unique and hopefully mnemonic names for each voice website. That Voice Portals fail to satisfy the Voice Web principles does not insinuate that the portal-centric market is naturally inferior – in fact, it may very well prove to be the more durable model – but the industry should not fail to distinguish between the two.

1.1.2. Focus

While the Voice Web is only one cross-section of the Web, it is difficult to resist the temptation to explore all the related avenues of market and technology development. Both the voice and Internet industries have a long and varied legacy that is brought to bear on this nascent market, and the sheer heterogeneity of products, services, and business models that they have spawned resist any attempt at generalization. In order to attend to the overarching question - what is required to ensure the realization of the Voice Web – the focus needs to be suitably sharpened. To this end, the author has adopted several assumptions and perspectives in the development of this paper.

Voice Web as Forward-Looking Opportunity. Until recently, the majority of the voice industry has been driven by the premise that speech-interactive applications can reduce the hassle and cost of maintaining a large staff, e.g. United Airline's automated flight arrival and departure information virtually eliminates the need for live operators to provide that information. While there are certainly a few notable exceptions such as unified messaging/personal assistant systems and even voice portals, the industry is highly targeted to serving companies that are simply trying to cut the fat out of their operational budget without jeopardizing their valued customer relationships. There are certainly many businesses that open shop on the HTML Web for the very same reason, e.g. United Airlines provides bonus frequent flyer miles to encourage consumers to make reservations online and thereby further reduce the operator costs. These kinds of businesses do not view Web or voice applications as an opportunity to create a new product, but as a means to cut costs for an existing product: in this case, United Airlines' products are airline tickets, and they would much prefer to automate part or all of the customer interface. In other words, the Voice Web is not a "forward-looking" business opportunity, but a means to cut the fat out of the current business.

For these kinds of businesses, the Voice Web is an interesting but largely irrelevant development in the voice industry. The increased standardization and popularization of the Voice Web will certainly lead to marginally lower development and hosting costs, but these pale in comparison to the cost savings afforded by automating the call in the first place. It should be appreciated these "fat-cutting" companies have always had sufficient incentive to make this investment, even before the emergence of the consumer voice portals or recent standardization efforts. They don't mind having a toll-free number since they have always had one, and they can easily partner with the existing consumer portals by leasing keywords that trigger a call transfer to their existing number. It may very well be the case that the industry will evolve to the point where these types of "inward-looking" companies will demand Voice Web compliance from

their current vendors, but this will not be the short-term reality. No slight is intended by the term "inward-looking" since these types of businesses will very likely continue to be the "cash cow" for the entire voice industry, especially within the virtually untapped small-to-medium size enterprise market.

While many of the analytical concepts and market observations developed in this paper will be of general interest and relevance to the "inward-looking" industry, the focus is on those firms that would otherwise eschew publishing a voice application. For these "forward-looking" businesses, the Web represents an opportunity to extend brand, develop new products, and enable commerce. For example, the consumer auction industry would not be possible without the existence of firms like eBay, leading destinations like Yahoo and CNN attract significant advertising revenues, and companies like Amazon have redefined the concept of impulse consumerism. For forward-looking companies like these, the website is not a cost-cutting measure to reduce its workforce – the website is the product. These types of firms would simply not exist were it not for the Web, and any effort to attract them to the Voice Web space must be consistent with the principles outlined earlier: one Web, standardized, fair and intuitive access, etc. The recent crop of voice portals are just the first iteration of many that are borne out of the forward-looking models of the Internet and lure of distribution over the largest network in the world: the telephone network.

The fraction of the voice market currently engaged in these types of forward-looking applications is admittedly small, and the author would be remiss in prognosticating any growth trend; however, this paper is focused on examining the technology and market implications for these types of market participants. It should also be noted that these types of business models are not mutually exclusive since a single site can reduce agent costs while enabling new services, e.g. the Land's End website not only supports online ordering of catalog items but also has an interactive application that allow consumers to create and custom outfits on a virtual model. The Voice Web model is certainly supportive of both these types of business models, but the value proposition is clearly more compelling for those forward-looking firms that would normally reject a voice application approach.

Mobile Telephony Platform as Launchpad. While the Voice Web itself places few restrictions on the type of end user device, the analysis would be unnecessarily complicated if every access technology were given equal consideration. Those enterprises for which the Voice Web is complementary to their traditional Web initiatives and represents a forward-looking publishing opportunity, mobile telephony is the clear choice for the initial deployment platform. While there will most certainly be radical changes in the wireless industry due to increasing packetization of the voicestream and popularity of more general devices like personal digital assistants, the pre-existing cellular consumer base is too large to ignore.

This mobile focus does not negate the ability of wireline telephones to participate on the Voice Web, but it does suggest a diminished role for traditional telephone in the general evolution of the industry. The inherent modularity and utility of the cellular industry encourages consumers to constantly upgrade their mobile devices to take advantage of new features: web browsing, paging, voice-activated dialing, phone book, even integrated music players. In contrast, the wireline telephone has changed little since its inception decades ago, with both businesses and consumers channeling their financial energies to multipurpose devices like computers. Unless one believes that the Voice Web will forever remain an isolated island whose input and output interface and characteristics will never change, then one must look to the cellular handset manufacturers and network operators to shepherd the evolution of the Voice Web. For example, traditional telephony is incapable of supporting stereo audio and visual output, which would present an insuperable barrier to the Voice Web's evolution were it not for the cellular industry.

This mobile focus also permits the author to forgo analysis of computer-based applications such as dictation, speech-interactive web browsing, etc. While they play a strategic role in the long-term development of the voice industry (e.g. Microsoft can popularize speech interaction through its desktop Office suite and progressively deploy the technology on its mobile operating systems), the technology discussion will be limited to the more immediate concerns of the mobile telephony infrastructure.

This paper does not sound the death knell for the traditional telephone since it will most certainly continue to be a popular access point for telephony and speech applications, particularly for inward-looking products. However, the author fundamentally believes that while the development path of the Voice Web must begin with telephony, it must not fail to move away from that heritage to embrace other modes of interaction. Even though many of the observations in this paper are equally relevant to wireline carriers and applications, much of the discussion will unapologetically focus on the enabling potential of the wireless industry.

Voice Web is Just Another Facet of the World Wide Web. Closely related to the presumption of mobile telephony as the initial platform is the mandate to move away from thinking about the Voice Web as its own entity rather than part of a much larger Web organism. This not only implies that consumers will demand more functionality from their voice applications, above and beyond that which can be provided by the current telephony network, but also that these applications will consider speech interaction as a commodity input element along with mouseclicks and keystrokes. For forward-looking firms, Voice Web applications will be seen as just another interface point to an existing application or

data set. In the long-term, it may be difficult to characterize Voice Web applications as stand-alone development projects. This may take the form of publishing through dynamic generation of markup code or a single language that supports multimodal publishing over a variety of platforms; regardless, the author's analysis of the Voice Web is from the perspective of the larger Web industry.

1.2. Strategic Analysis Methodology

In order to assess the necessary and sufficient conditions for a sustainable Voice Web, this paper serves as a *strategic analysis* of the Voice Web industry, developed in two stages:

1.2.1. Structural Analysis

This initial step involves partitioning the relevant market into atomic functional units and enumerating the array of forces that drive industry competition. Since the Voice Web is an industry and not simply an engineering construct, such an analysis is inherently multidisciplinary in nature. Each market partition is subject to variety of factors, the importance of which depends on the market segment under examination.

- **Technological**. Is the existing Internet capable of supporting the quality of service expected of Voice Web applications?
- **Economic**. If a carrier's cellular network is already being strained by the commuter load, should that carrier deploy a Voice Web service that demonstrably encourages long-hold times during those same peak hours?
- **Regulatory**. Does intellectual property infringement occur when audio content from the HTML Web is converted for playback over the telephone system?
- **Human**. How do consumers indicate a Voice Web site address if there is no centralized, objective assignment of mnemonic keywords?

In the context of the Voice Web, the first two factors play a much more significant role than the latter two. The speech and Internet industries are certainly subject to varying degrees of regulatory oversight, but only indirectly; predominantly non-governmental standards bodies are relied upon to govern industry-wide cooperation and conduct. And while human factors (e.g. usability) obviously play a key role in the success of any particular application, the non-application-specific observations are notable but far less in number. A Web model must allow for poor design and rely on the competitive market to provide

sufficient incentive for superior, human factors-driven design. This does not imply that regulatory and human factors will be omitted from the analysis, but that the readers' expectations should anticipate an investigation that focuses heavily on technological and economic factors. Figure **Error! Not a valid link.** below illustrates the iterative structural analysis methodology:

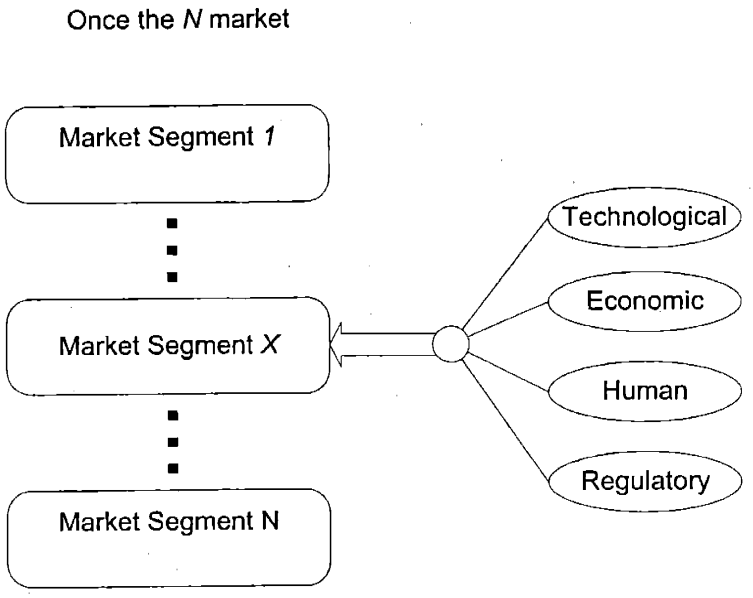

Figure 2. Structural Analysis Methodology

1.2.2. Scenario Analysis

This second stage of the investigation involves examining the results of structural analysis – market segments and attendant factors – and exploring the potential market configurations that both encourage and discourage the development of the Voice Web. In contrast to the iterative structural analysis that considers each market segment in isolation, scenario analysis focuses the construction of the entire market from these constituent elements. For example, if the Voice Web model tends to commoditize previously distinct elements of the market, then the market may respond by a shift in segment configuration: vertical disintegration coupled with cost leadership, accelerated vertical integration with differentiated elements, narrowed market focus on niche applications, merger with former supplier or competitor. These market configurations or scenarios are not necessarily mutually exclusive, e.g. AOL's closed model of the consumer online experience coexists and even thrives alongside the HTML Web industry. As the scenario analysis is extended forward in time, the tension of what is likely to occur versus what needs to occur to ensure the Voice Web model becomes more apparent. While the investigation is conducted from the

viewpoint of the entire voice industry, the analysis is bolstered with examples from specific firms in the speech and Internet industries.

1.2.3. The Role and Scope of Strategic Analysis

From the perspective of a single firm, the goal of strategic analysis is to find a position in the industry where the company can best defend itself against competitive forces or influence them in its favor. They key is to delve below the surface and analyze the sources of these competitive forces, the knowledge of which highlights the firm's critical strengths and weaknesses, animates the positioning in its industry, clarifies the areas where strategic chances may yield the greatest payoff, and underscores industry trends that hold the greatest significance. The fact that the Voice Web industry under examination does not yet exist further complicates the analysis simply because there are no governing rules of the game, the absence of which is both a risk and an opportunity to be managed.[7] Strategic analysis is certainly no substitute for intuition, but it compels the investigator to systematically examine the interrelationships and forces within the market. In a sense, this type of analysis couches intuition within a more formal framework and discourages short-sighted and ill-considered proposals.

While the industry-wide structural and scenario analyses developed in this paper are certainly part and parcel of any competitive strategy formulation, they serve only as the starting point for any individual firm. The creation of a sustainable Voice Web is a matter of market configuration, and not necessarily the natural result of long-term profit maximization for any single firm. A thorough strategic analysis compels each company to further examine the forces acting upon it specifically: bargaining power of suppliers and buyers, barrier to new entrants, threat of substitute products or services, rivalry among existing competitors, liquidity of financial assets, expertise and capability of existing management, etc. While this paper can stimulate or inform further investigations, it is no surrogate for a firm's own strategic analysis.

While the Voice Web is a unique vision of the voice industry, many of the determinants for its success are held in common with other market models, e.g. both the Voice Web and Voice Portal visions would profit greatly from improved speech recognition, lowered recurring telephony costs, higher fidelity on audio output, etc. The reader should not be surprised that much of the discussion is applicable beyond the strict Voice Web construct; indeed, it could be argued that the Voice Portal and other non-Web models incorporate the necessary but not sufficient conditions for the Voice Web.

For as much as a strategic analysis can provide, it is equally important to appreciate what it cannot offer. No proof is provided that the Voice Web would ultimately be better for maximizing market efficiency, nor any argument that anything short of the Voice Web would have a deleterious effect on the economy. Furthermore, this investigation does not address the issue of consumer demand or acceptance, e.g. will people find certain Voice Web applications valuable enough to spend their money on, particularly in the context of well-entrenched substitutes like radio, television, and the Internet? It does not suggest the types of applications that would be successful or how best to implement them. Perhaps most importantly, a high-level strategic analysis does a poor job of predicting the seemingly irrational behavior of market participants, fueled primarily by emotional attachment (e.g. the proverbial triumph of hope over experience) and the failure to consider competitive strategic analysis as an indispensable element of any enterprise.

1.2.4. The WML Web Case

In order to motivate this systematic and interdisciplinary consideration of the voice market, consider the apparent lackluster performance of the Wireless Application Protocol (WAP) industry. The suite of protocol and document (e.g. Wireless Markup Language - WML) standards defined by the WAP Forum,[8] coupled with the enthusiastic adoption by the handset manufacturers and wireless service providers, form the backbone of the WML Web. While it is entirely premature to pronounce the death of cell phone mini-screen access to the Web, the initial perception (whether reflecting the true sentiment of the consumers or the enclave of industry analysts) has led to a recent addition to the Web lexicon: *WAPapthy* (WAP apathy).[9] However exasperating to the consumer, the customary criticisms of slow connect speeds and lackluster content are not the result of a deficiency in the WAP standards themselves, since both can be addressed by increased competition in the applications space and the inevitable evolution of the cellular data network.

---

[7] Michael E. Porter's seminal contribution, *Competitive Strategy: Techniques for Analyzing Industries and Competitors*, provides a thorough and approachable introduction to the discipline of strategic analysis.
[8] The WAP Forum is the industry association comprising over 500 members that has developed the *de facto* standard (WML) for visual wireless information on digital mobile phones. See http://www.wapforum.org/.
[9] "Most people have barely heard of WAP phones - the mobiles which use a cut-down version of the Internet - yet already there's a new word to get used to: WAPathy. Instead of connecting to the latest technology, the early signs are that people are switching off." Patrick Collinson, "Phones Fail to Ring Up Sales," Guardian Unlimited, 8 July 2000. See: http://www.guardian.co.uk/Archive/Article/0,4273,4038212,00.html.

These peripheral concerns divert attention from the more fundamental, market-defining forces within the WML Web. This chapter's opening quote, admonishing Voice Web observers to look past the hackneyed references to the traditional Web to identify and overcome the critical differences, holds true on many levels for screen phone services. For example, since the WML and HTML technology architectures so closely resemble each other, the assumption was that the WAP industry would evolve along lines similar to the HTML Web; however, the technological similarity should not mask the importance of the other factors.

- **Regulatory**. The cellular carriers are not subject to the same common carrier rules as their wireline counterparts. For example, whereas local telephone companies must allow consumers to choose any long distance carrier and Internet Service provider, the cellular carriers are under no such obligation and consequently offer a single, integrated product. This regulatory asymmetry implies a market structure dictated by the parochial interests of the cellular carriers instead of the entire HTML Web community.
- **Economic**. Prior to the introduction of WAP services, the cellular industry already engaged in subsidization of the handset costs, ensuring that cellphones would work only a single carrier's network. Given the unusually tight relationship between the carriers and handset manufacturers, the browsers are developed in such a way that the start page is immutable, i.e. a subscriber to carrier X is compelled to begin each WAP "surfing" experience with carrier X's start page. Again, though the HTML browser antecedent would suggest a user-configurable start or "home" page, the economic forces guarantee an WML browser that is captive to the carrier's portal; indeed, the portal site itself and the WML browser are difficult to distinguish from the consumer perspective.
- **Human**. Given a captive portal audience, the cellular carrier can rely on the user interface itself to create yet another revenue stream: portal placements. Since it is so cumbersome to enter a full Web address using only the keypad on the phone, the vast majority of users simply rely on the menu choices provided on the carrier's default page. This fact is not lost on the carriers that charge large sums for top spots on the default menu, reducing the effective WML Web field to a certain few with the financial wherewithal to secure a position.[10]

In a very real sense, the previously separate elements of end-user device, access, browser, and portal are rolled into a single integrated service. If the vision of the WML Web is to achieve virtues similar to those ascribed to the Voice Web – one WML Web, fair and democratic access, etc. – then WAP has not and

[10] InformationWeek.com, News, Carrier Strategy Limits WAP Phone Users' Online Choices, 30 October 2000. See http://www.informationweek.com/810/wireap2.htm.

will likely never live up to this expectation. A Web is not simply a technology construct but an entire market, and the presence of economic, regulatory, and human factors dramatically affect the evolution of this industry. Unlike the issues of underwhelming speed and application selection, these factors are not short-lived: the proponents of the WML Web that adopt a wait-and-see attitude fail to recognize these fundamental barriers created by the existing market structure. While a thriving industry may indeed develop around these carrier-specific WML portals, it should not be confused with a WML Web.

1.3. Paper Organization

As the WML Web case above demonstrates, market structure and factors can dramatically affect the evolution of a new product, despite frequent allusions to the well-understood HTML Web model. Since one might suspect that the Voice Web's existence is similarly contingent upon these elements, this paper develops a strategic analysis of this embryonic industry and presents a few key findings that will likely govern its evolution. With the central goal of identifying the necessary and sufficient conditions for the Voice Web, the paper is developed as follows:

Part I. Technology Primer

Before embarking on the strategic analysis, the technological foundations for the Voice Web are established. This does not suggest the precedence of technology factors over all others, but simply reflects the general level of understanding of speech-interactive technologies. Unlike traditional Web infrastructural elements such as servers and routers, most people, even within the engineering community, are unfamiliar with technologies such as automatic speech recognition and text-to-speech synthesis. Conversely, many within the voice community itself are unfamiliar with the client-server paradigm and the implications for speech application development and deployment. This section not only provides an overview of the intersection of HTML Web and speech technologies, but also an introduction to the equally important telephony and data networks to which they attach. The reader must first thoroughly familiarize herself with these technology concepts in order to appreciate and understand the following strategic analysis.

Part II. Strategic Analysis

Structural Analysis. The voice industry will be partitioned into its constituent market segments, starting with content providers and ending with consumers, the ultimate arbiters of the Voice Web's success. Given this particular decomposition of the market, each segment will be examined for the presence or absence of factors necessary to the creation of the Voice Web. While the analysis shall be drawn from the full palette of technological, economic, regulatory, and human factors, the discussion may track to a certain factor by virtue of the segment's definition. For example, while audio content on the Voice Web is potentially subject to the same regulatory oversight as broadcast content (e.g. censorship), the net effect on market development is negligible.[11]

Scenario Analysis. Given the segments and factors defined above, the various market configurations are explored for a variety of time horizons. For example, natural points of aggregation and disaggregation are highlighted and probable integration strategies are outlined. The analysis is interspersed with examples of individual firms to motivate the importance and relevance of the arguments developed in this section.

Conclusions

Given the length of this paper, the barrage of details may distract the reader from keeping the ultimate goal in mind: the conditions for the creation of the Voice Web. In order to reinforce the overall organization and flow of the analysis, Figure **Error! Not a valid link.** below will be frequently employed throughout this paper as a visual, contextual cue:

[11] Perhaps the most legendary example of broadcast censorship is the story behind comedian George Carlin's "Seven Dirty Words." When his "Filthy Words" monologue was played on the air in October 1973 by Pacifica radio station WBAI, it resulted in an order from the Federal Communications Commission (FCC) forbidding the broadcast of such language "at times of day when there is a reasonable risk that children may be in the audience." See http://www.beacham.com/seven_words.html.

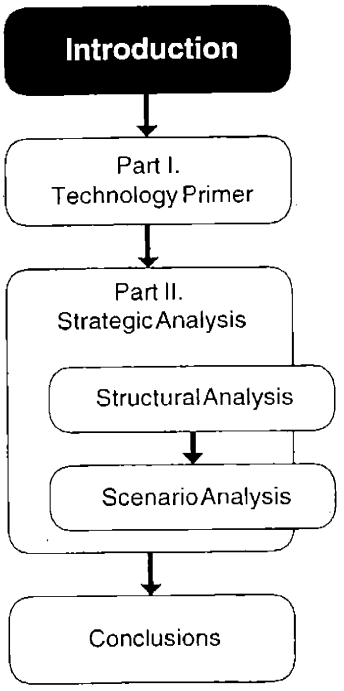

Figure 3. Overview of Paper (Currently on Introduction)

PART I. TECHNOLOGY PRIMER

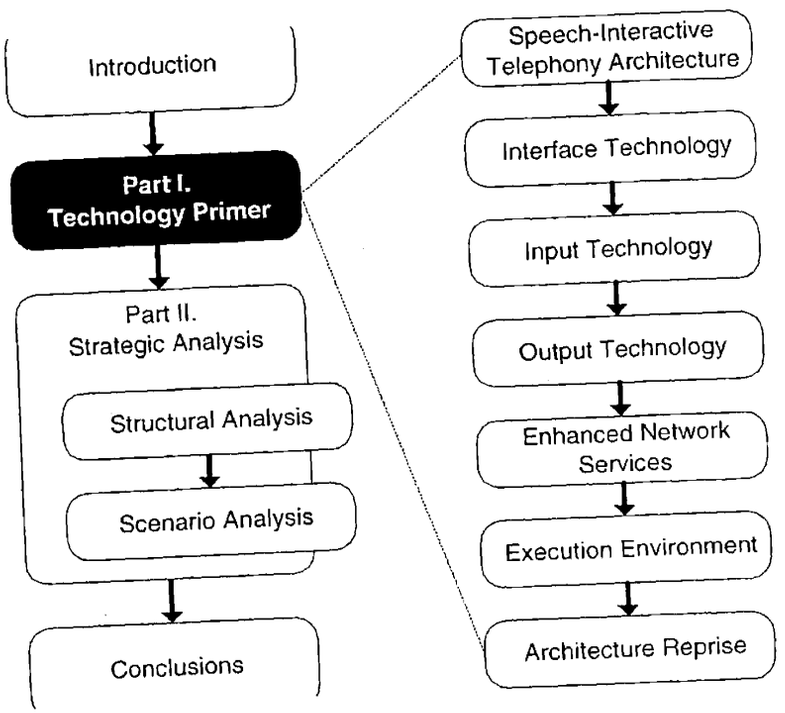

Figure 4. Overview of Paper (Currently on Technology Primer)

"A little inaccuracy sometimes saves a ton of explanation."

- H. H. Munro (Saki) (1870 – 1916)

The centerpiece of this paper is the strategic analysis of the Voice Web, so this primer serves only as a backdrop against which this investigation is conducted. Given the general unfamiliarity with voice applications and the recent introduction of the client-server application model to this industry, this Technology Primer explores the various components of the Speech Interactive Telephony Architecture: interfaces, input and output technologies, enhanced network services, and the execution environment. Before leaving this technology briefing, the reader will reexamine the architecture in the light of the more granular knowledge of each element.

As the quote above suggests, simplifications have been made throughout to abbreviate this preliminary material without sacrificing the important facets of each element.

Chapter 2. SPEECH-INTERACTIVE TELEPHONY ARCHITECTURE

> "The success of automatic speech recognition depends critically on the specific task to which it is put... witness the toy dog "Rex" of yore who wagged his tail in recognition when addressed [by name]. According to John Pierce, my longtime boss at Bell, this was – and he believed would remain – the only useful application of speech recognition."[12]

2.1. Brief History of Voice Applications[13]

As the name would suggest, the distinguishing interface element of the Voice Web is the capacity to interpret human speech. Despite the availability of telephony systems that rely on keypad touch tones for over half a century, the consumer attraction to and superior flexibility of the voice user interface is the driving force behind the Voice Web and other voice industries.

While commercial speech technologies have been available for almost 2 decades, they have only recently achieved notable market success and consumer awareness. Like the Internet itself, the beginnings of automated speech recognition (ASR) can be traced to a series of projects in the 1970s sponsored by the United States Defense Advanced Research Projects Agency (DARPA). DARPA's Speech Understanding Research (SUR) program spent $15 million over 5 years to develop machines that could recognize very constrained speech from a limited number of speakers. The most successful system, named HARPY, was developed at Carnegie Mellon University (CMU) and was able to recognize complete sentences that were constrained to a very limited grammar. The need to excessively constrain user input and the extraordinary expense of computing resources (HARPY required 50 DEC PDP-11 computers running in parallel for a single channel of recognition) relegated ASR technology to the realm of academic curiosity for many years.

As computing resources became increasingly available to large enterprises, vertical applications began to appear in certain industries, e.g. digit recognition to retrieve account balances. Enterprise systems that

---

[12] Excerpted from "Speech Processing" by Manfred R. Schroder, Signal Processing for Multimedia, Volume 129, J.S. Byrnes (Ed.), IOS Press, 1999. See: http://www.cs.umb.edu/~asi/multimedia/papers/schroeder.pdf.

[13] This section borrows heavily from the following two sources: NetByTel eGram Industry Newsletter, Issue 3, September 2000 (See: http://www.netbytel.com/literature/e-gram/technical3.htm), SpeechWorks Technology Backgrounder (See: http://www.speechworks.com/learn/essentials/guide_pdfs/tech_backgrounder.pdf).

implemented these constrained applications were costly to build, but using human operators to address routine questions were even more costly, so the cost of custom hardware and programming was justified.

The 1990s marked the beginnings of unprecedented technological and financial growth for the voice application industry. In 1995, Dragon Systems released the first dictation program available for the consumer market, with IBM and Kurzweil following close on it heels. In 1996, Charles Schwab became one of the first companies to support a large-scale speech recognition application – a quotation service for stocks and options - the success of which set the stage for other large enterprises to adopt this model of customer service. In that same year, BellSouth launched the world's first voice portal, called Val and later renamed Info By Voice.

The appearance of speech recognition systems on general purpose computer systems, as well as software tools aimed at rapid application development by the end user, have created the possibility of incorporating speech-interactive applications into even the smallest of enterprises, and the focus of the voice industry remains firmly focused on the automation of repetitive enterprise tasks. The engineering strides that have enabled the flourishing of the enterprise market have also made possible the cost-effective deployment of large-scale consumer services, such as voice-activated dialing and voice portal services.

As this Technology Primer will demonstrate, the architecture required to support voice applications involves much more than the mechanics of speech recognition. The novelty of this technology tends to polarize the focus to the input interface and leaves unexamined the system within which it operates. The following discussion encourages a more "holistic" approach to understanding speech-interactive architectures since the creation of the Voice Web relies on a variety of non-speech-specific technologies.

2.2. Reference Model for Application Interaction

Whether a simple "Hello, World" program written in BASIC or an e-commerce website, program execution can be decomposed into the following basic components:

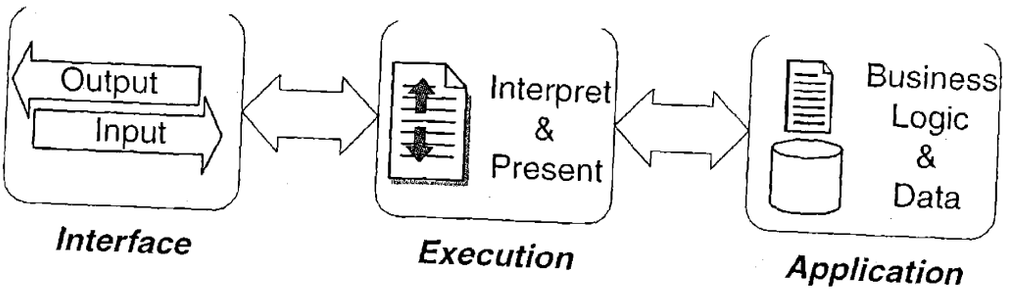

Figure 5. Reference Model for Application Interaction

It is important to note that the diagram implies nothing about the types, locations, or even number of devices necessary to implement a program. For example, in the case of the aforementioned BASIC program:

- **Interface**. The input device is the keyboard, and the output device is the monitor.
- **Execution**. The computer has a run-time environment that can both execute machine code and manage the attached I/O devices.
- **Application**. The compiled code for this simple program is stored locally in memory.

In the BASIC example, all three components reside in the same geographic location and are physically integrated within a single computer system. In the case of the e-commerce system:

- **Interface**. In a typical surfing application, the input devices are the keyboard and mouse, and the output devices are the monitor and sometimes the speakers.
- **Execute**[14]. An HTML-aware browser is installed on the computer to which the I/O devices are attached.
- **Application**. The HTML documents, relevant graphic and text files, and business logic are stored remotely across the Internet on the service provider's servers.

Unlike the simple BASIC program, there are two organizational entities involved, separated geographically by a public data network, but the abstraction still holds. This metaphor for program interaction will illustrate the essential difference between Visual and Voice Web architectures.

2.3. Consumer-Side vs. Server Provider-Side Execution

[14] Even though the word *execute* typically connotes the running of compiled machine code, it is used here to also cover the run-time interpretation of markup languages.

In both of the examples above, the execution function was performed on a machine physically proximal to the end user, but this need not be the case for all program interactions; instead, the service provider itself can assume this responsibility on the user's behalf. Another element can be added to the reference model to capture this distinction: organizational boundaries.

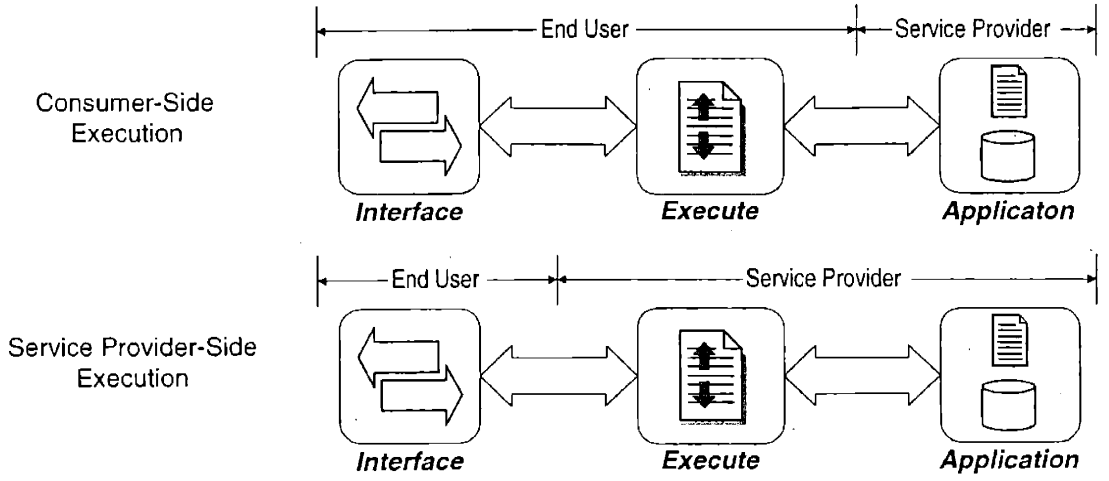

Figure 6. Consumer-Side vs. Server Provider-Side Execution

While almost all visual web models, including the traditional HTML Web and the more recent WML Web, employ *consumer-side* code execution, most architectures designed for speech-interactive telephony assume the *service provider-side model*. Most telephones lack the computational resources for speech recognition and audio decoding but already possesses the necessary interface technologies (i.e. monophonic microphone and speaker), so the remote situation of the execution technology is the only viable short-term solution for enabling speech-interactive telephony.

2.4. Applying the Reference Model

2.4.1. HTML Web

It should be remembered that the reference model represents logical, not physical data flow, so each module may in fact be a collection of devices or the entire flow may occur within a single machine. Expanding the model for the specifics of the visual and voice architectures will further illustrate the distinction.

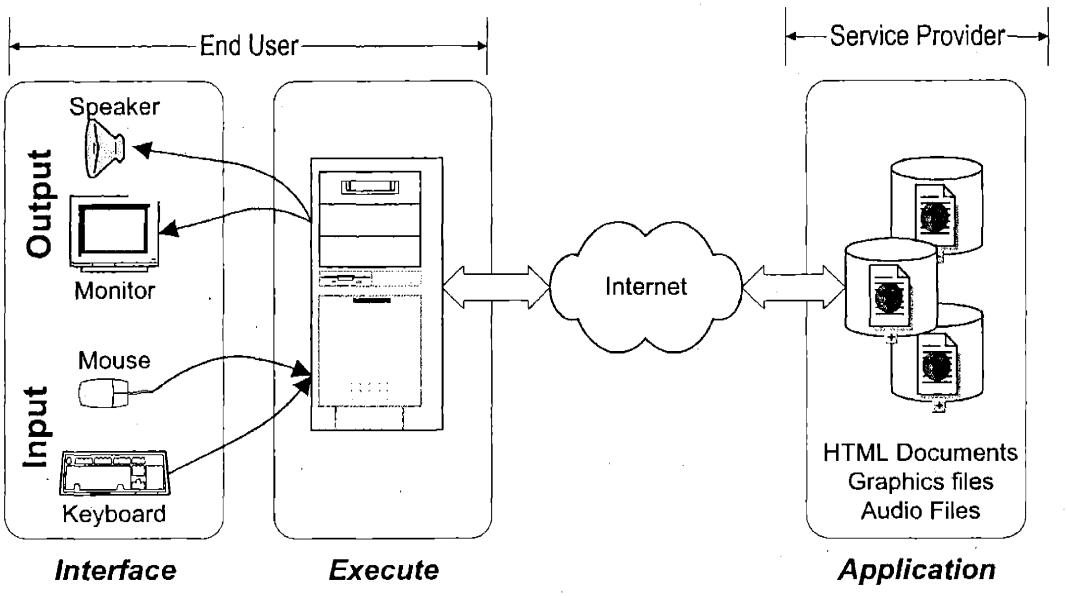

Figure 7. Applying the Reference Model to the HTML Web

Instead of retrieving the data locally for execution, the markup documents and related files are remotely located in a typical HTML Web session. While many organizations are involved in conveying Internet Protocol (IP) packets across the Internet, they are not identified as HTML service providers. Bit conveyance is considered a commodity service to which end users and application providers subscribe, and any site on the Internet network is generally accessible to any other connected device. The fortuitous combination of a public document standard (HTML) and a ubiquitous, fully-interconnected Internet has resulted in overwhelming adoption by the content, product, and service industries. Since the Internet is built upon a best-effort packet delivery paradigm and quality-of-service (QoS) mechanisms have not been widely deployed throughout the network, Internet traffic is subject to variable latency and throughput. As a result, the Internet has been employed with great success for publishing non-time critical content, but is generally unsuitable for real-time applications.[15]

2.4.2. WML Web

While the consumer perception of and experience with the WML Web may be dramatically different from that of the HTML Web, the architecture style is fundamentally similar: consumer-side execution.

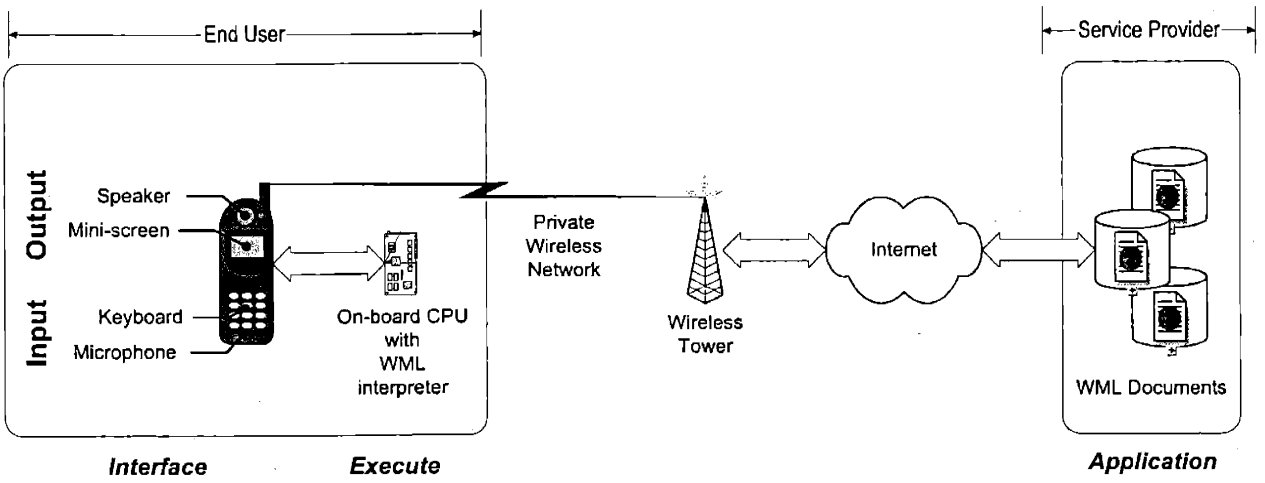

Figure 8. Applying the Reference Model to the WML Web

The two differences from the HTML example, apart from the document standard itself, are the devices involved and the addition of a private (wireless) network. The cadre of I/O devices and even the interpreter itself are replaced by a single cellular telephone: interface and execution tightly integrated into a single end user device. Instead of being tethered directly to the Internet, the phone communicates through a wireless network operated by a cellular carrier. Since WML documents traverse the same public network as do HTML pages, applications are designed with variable latency and throughput in mind-the wireless leg often the slowest link in the flow from handset to service provider. Despite the differences in protocols and document formats, other wireless handheld devices (e.g. Palm Pilot) that support visual applications rely on the same consumer-side execution model.

2.4.3. Speech-Interactive Telephony

Whereas HTML and WML applications embrace consumer-side execution, speech-interactive telephony (SIT) architectures traditionally feature *service provider-side execution.*[16]

[15] Live content can be streamed over the Internet, but at the cost of significantly reduced fidelity and frequent interruptions.

[16] The author has coined the term *speech-interactive telephony* to avoid using the more common term *interactive voice response* (IVR). Given the long and varied history of the voice industry, the term *IVR* often applies to systems that support only touchtone access and it does not reinforce the connection to the telephony network.

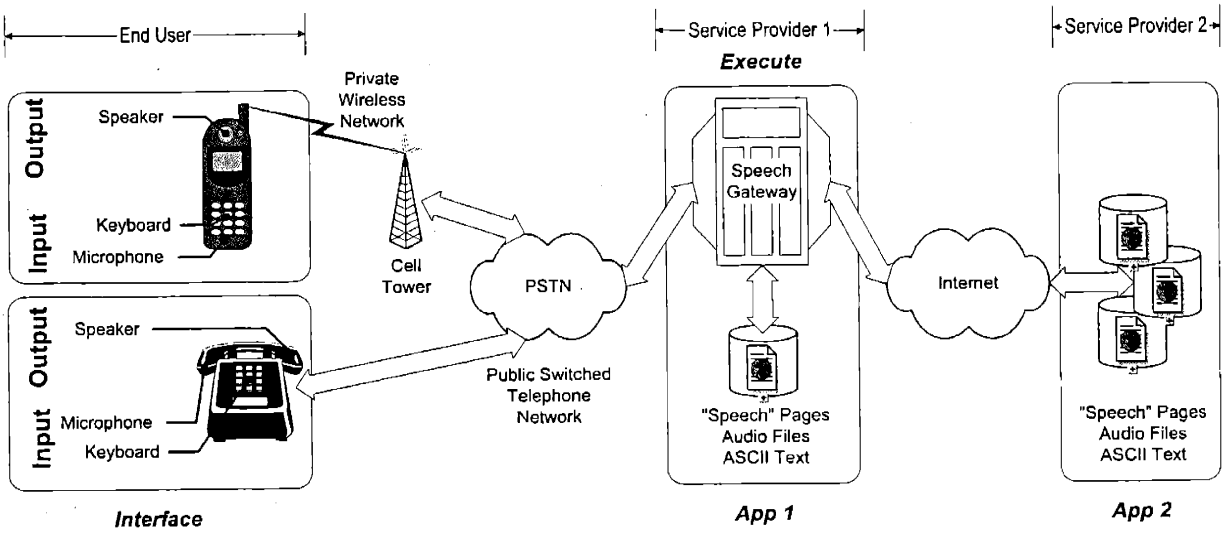

Figure 9. Applying the Reference Model to Speech-Interactive Telephony

Unlike WML-based services that involve the purchase of an enhanced cellular phone, SIT applications require nothing more than a traditional wireline or wireless telephone. SIT services rely on nothing more than the phone network's ability to send and receive low-fidelity audio signals in real-time.[17] While the local access portions of the cellular and tethered networks employ different technologies, most of the calls eventually pass through the *Public Switched Telephone Network* (PSTN).

Unlike the packet-switched Internet, the PSTN employs physical or virtual circuit technologies: resources are dedicated to each call to ensure high quality of service throughout the call duration. While the Internet serves as a generic bit conveyance network that supports a variety of protocols and applications, the PSTN has been engineered specifically to carry bi-directional human voice conversations in the form of 64-Kbps monophonic audio signals. Similar to the Internet, the PSTN is comprised of many interconnected private organizations, ensuring that any telephony device can generally reach any other device on the network (i.e. near-universal access to and through the network). In order to appreciate the

[17] Even a computer capable of Internet telephony would suffice, so long as the quality of service could be maintained and it could place a call on the public telephone network. This issue of service quality will resurface later: wireless networks tend to distort the audio signal more than do wireline networks, exacerbating the speech recognition process.

scale of this network, as of July 2000, there were approximately 192 million wireline[18] and 97 million wireless[19] in the United States alone, as opposed to 93 million Internet hosts worldwide.[20]

Regardless of its origin, the call terminates at the *speech gateway*, a collection of technologies and devices that performs of variety of call processing functions to support the SIT application: call management, speech recognition, audio playback, computerized reading of ASCII text, access to data storage. Most SIT applications have traditionally been deployed in situations where the program structure is quite static and variable data, if any, is accessible from within the enterprises' own data network. This case of a completely "self-contained" SIT application is denoted by the portion of Figure 9 bracketed by "Service Provider 1": both execution and data occur within the confines of a single organization.

With the advent of business models that generally decouple the speech gateway and the application – voice portal that relies on external content, voice hosting of remotely generated speech programs, and most notably, the Voice Web - the data model must be extended to include application elements located across physical and organizational boundaries. While Service Provider 1 retains *execution* responsibility vis-à-vis the speech gateway, the *application* service provider (labeled "Service Provider 2" in Figure 9) can flexibly distribute its applications on the same LAN segment, across the Internet, etc.[21] In the industry vernacular, Service Provider 1 is called the *Voice Service Provider* (VSP) and Service Provider 2 is given the moniker *Application Service Provider* (ASP). In a sense, the VSP is the speech analog of the Internet Service Provider (ISP), providing commodity "speech conveyance" instead of a bit delivery service. A single organization can certainly encompass both roles (e.g. self-contained SIT application), but the VSP and ASP functions remain distinct.

2.5. Speech Gateway

[18] Over 100 million of the 192 million total are residential consumers. Federal Communications Commission Biannual *Trends in Telephone Service*, June 2000 sample (http://www.fcc.gov/Bureaus/Common_Carrier/Reports/FCC-State_Link/IAD/trend200.pdf). See also Federal Communications Commission *Telephone Subscribership in the United States*, November 2000 sample (http://www.fcc.gov/Bureaus/Common_Carrier/Reports/FCC-State_Link/IAD/subs1100.pdf).

[19] There are approximately 46,000 new wireless subscribers every day: one every two seconds. Cellular Telecommunications & Internet Association Semi-Annual *Wireless Industry Survey*, June 2000 sample (http://www.wow-com.com/statsurv/survey/data.cfm).

[20] Internet Software Consortium Semi-Annual *Domain Survey*, July 2000 sample (http://www.isc.org/ds/).

[21] *Consumer-side interpretation* is prevalently termed *client-side*, and *service provider-side* as simply *server-side*. The author eschews the more popular expressions because of the inherent lack of precision and potential for confusion, e.g. since a *client* can be variously defined as the end user device or the device responsible for markup language interpretation, is the telephone or the speech gateway the *client* in the SIT architecture?

While the SIT architecture employs elements mostly borrowed from other networks (e.g. telephones, Internet, PSTN, servers, databases), the speech gateway is unique to this type of network. Despite its central importance, most readers are unfamiliar with the functionality and composition of this network element, but such familiarity is central to understanding the current voice market and its evolutionary course. More so than any other component in the voice architecture, the speech gateway will dramatically and specifically shape the nature of the Voice Web, including whether it will be achieved at all. Only when this causal relationship between technology and market structure is appreciated can a positive (what will be) and normative (what should be) analysis be conducted.

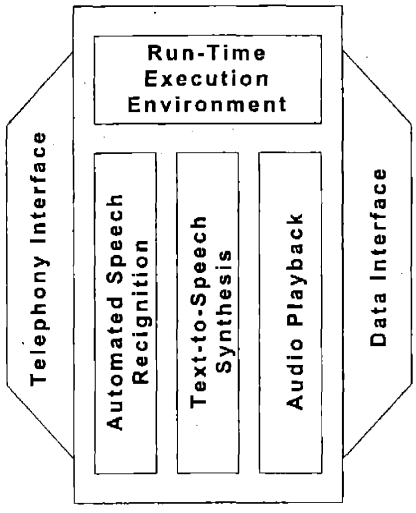

Figure 10. Constituent Components of the Speech Gateway

As the term *service provider-side execution* would suggest, the speech gateway's role is to interpret and present the data (program logic, content, etc.), which involves a variety of technologies in addition to speech recognition:

- **Run-Time Execution Environment**. In general, all of the interfaces and functional components act as "slave" processes to this "master" process. Once the telephony interface signals the arrival of a new call, a session construct is created and the application begins execution. Depending on the language type, this environment either executes pre-compiled code or interprets markup language. Most functional aspects of the user *voicestream* (speech analog of *clickstream*) and the system performance can be observed from this vantage point.
- **Telephony and Data Interfaces.** These modules are responsible for engaging the PSTN and IP-based networks, respectively. The telephony interface dedicates DSP resources to each circuit

throughout the duration of the call, whereas the data interface provides packet-switched access to both local intranets and the larger public Internet.

- **Automated Speech Recognition.** Available in both software and hardware formats, this component is ultimately responsible for turning the caller's spoken directives into computer-interpretable ASCII values. Unlike a dictation program, SIT applications utilize speech recognition technologies that work without explicit "training" by the user.
- **Text-to-Speech Synthesis.** In some respects, this module performs the inverse operation of the ASR engine: converting ASCII text into computer generated "speech." While these systems do not currently produce audio that would be mistaken for real human speech, it provides an inexpensive and real-time means of utilizing any text-based content already available on the Web (which is arguably more plentiful than audio content). Like ASR systems, text-to-speech engines are available in both hardware and software implementations.
- **Audio Playback.** This module is responsible for modulating a digital representation of recorded sound into a PSTN-compatible signal. The types of audio files may be as compact as a terse welcome message, or as protracted as a continuous live feed from a radio station. Like the popular audio plug-ins to HTML browsers, the playback module provides the link between the audio files encountered in the execution environment (i.e. voice "browser") and the telephony interface.

It's important to note that this discussion provides a functional (e.g. logical), not physical description of the speech gateway. In other words, the speech gateway is not necessarily a single integrated device but potentially a network of component resources, especially for large installations in which economies of scale and fault tolerance can be gained by creating pools of ASR engines, telephony interfaces, etc. The HTML world provides a familiar example: a Web site is not a *device*, but the collective *action* or functional *sum* of the servers, databases, firewalls, LAN segments, element management systems, etc. This does not imply a loose confederation of devices because there is a complex, integrative value provided by the speech gateway vendor to ensure proper operation-it cannot be casually scraped together from third party components as if it were a hobby PC.

It should be remembered that the speech gateway hails from a long tradition of interactive voice response platforms: it is not the invention of the Voice Web. While there is no doubt that each component has profited from advances in both basic research and engineering gains, these improvements are largely a matter of performance not kind. The central technological novelty introduced by the Voice Web is the need to support the client-server paradigm of the Internet. This not only portends the introduction of

speech-centric markup languages, but also sets expectations for density, scale, performance, availability, formats, and application heterogeneity – all of which impact the traditional elements of the speech gateway and require the addition of new components. The following chapters explore these speech gateway functionalities in the critical light of the Voice Web: interfaces, input and output technologies, enhanced network services, execution environment.

Chapter 3. INTERFACE TECHNOLOGY - TELEPHONY & DATA

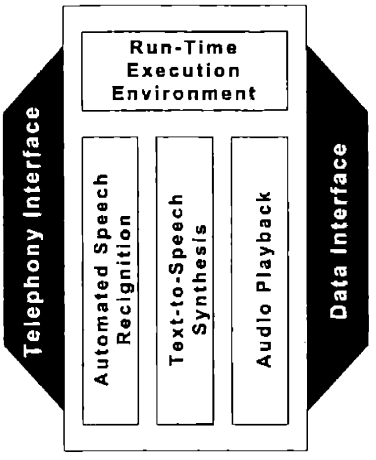

Figure 11. Speech Gateway (Currently on Interface Technology)

A speech gateway must interface with two very different networks: the PSTN and the Internet. The PSTN is a circuit-switched network that is optimized for a single service and provides reliable QoS; in contrast, the Internet employs a packet-switched design that supports a variety of services and offers no consistent QoS. The technology to interface to each of these networks is widely available, but the prospect of integrating both to deliver a single service is formidable given the obvious mismatch in network characteristics.

3.1. Conventional (Circuit-Switched) Telephony Interfaces

Telephony interfaces are employed widely in traditional IVR systems, call centers, enterprise PBXs, etc. In addition to simply "answering" the call in lieu of a person, the telephony interface card supports many of the ISDN and SS7 signaling features: call hold, conference call, call forwarding, simultaneous transfer and hold, extract ANI/DNIS[22] information, etc. In addition to "commodity" call servicing, boards optimized for SIT applications incorporate other features to improve the accuracy and scalability of

[22] *Automatic Number Identification* (ANI) and *Dialed Number Identification Service* (DNIS) provide information about the incoming call to the recipient, such as the number from which the person is calling and the number that was dialed. The familiar residential caller ID boxes use these services to provide the originating number. Speech gateways can use this information to pre-identify the caller based on the originating number, or they can vary the startup "page" of the speech application based on the number dialed (e.g. a single gateway could support both a restaurant application at 888-888-FOOD and a traffic application at 877-TRAFFIC).

speech recognition. *Echo cancellation* improves DTMF[23] or ASR performance by eliminating leakage of the outgoing signal into the receive signal path, an all-too-common experience for frustrated cell phone users who hear a garbled sum of the other caller and their own voice echoed back after a brief delay. *Endpointing* functionality continuously monitors the incoming audio signal for the beginning and ending of speech, which includes the rejection of silence and background noise as valid speech inputs. Endpointing increases efficiency and accuracy by first screening the incoming signal for a valid speech fragment, and only then forwarding it onto the ASR resources for subsequent processing. The telephony card is ultimately responsible for encoding and decoding the incoming and outgoing audio signals. Similar to a computer audio card's intermediary role between the audio files and the speakers, the telephony interface performs significant signal processing to ensure compatibility between the PSTN's and the speech gateway's internal audio protocols and formats.

3.1.1. Port Capacity

Unlike residential telephony networks that require a analog copper pair for each additional service line, large enterprises demand higher-capacity digital technologies to manage their call volume, so most employ T-carrier links like T1 (24 calls per circuit), T3 (720 calls per circuit), and even higher capacity technologies (e.g. a fiber optic OC3 can handle 2160 calls concurrently). In light of these advances in traffic density, telephony interfaces have similarly responded with improvements in port capacity. A *port* is simply the telephony interface counterpart to a PSTN *call*: it is the atomic (i.e. indivisible) unit of capacity. A port is both a physical and logical construct in that it provides both a material interface point and dedicated DSP resources. Following the circuit-switched convention of the PSTN, each port is uniquely and continuously assigned to a single call through its duration, so the number of ports on a speech gateway conveys its simultaneous call capacity and adheres to a strict one-to-one relationship.

Driven by the large-scale speech applications that utilize the larger digital carrier links above, the innovations in telephony interfaces have focused on port scalability (i.e. incremental and unconstrained capacity expansion) and density (i.e. high port count per interface element). Scalability is addressed by the ability to "pool" multiple cards to provide a single virtual interface, e.g. 10 cards with 48 ports per card provide an effective 480 port capacity. Improvements in density are driven primarily by more

[23] Dual Tone Multi Frequency (DTMF) is the technical term for describing push button or Touch Tone dialing (Touch Tone is a registered trademark of AT&T). The name derives from the fact the unique sound made by pushing each key is actually the combination of two tones, one high frequency and the other low frequency. For more details, see: http://www.voicesaver.com/glossary/g_dtmf.htm.

general advancements in the DSP industry; currently, the highest density cards that include the expanded ASR feature set (e.g. endpointing) support 4 T1s (96 ports) on a single card.

3.1.2. Audio Fidelity

Since the PSTN was ostensibly designed and optimized to carry human conversations, the engineering assumption of this traffic type has been inflexibly incorporated into all facets of the architecture. When this telephony-centric network is marshaled into providing other types of services, the mismatch of engineering assumptions creates friction. One notable example of this legacy is the inflexible and insuperable standard of audio fidelity: the PSTN network was designed to carry 8-kHz/8-bit/mono/μlaw PCM audio.

- **8-kHz Sampling Rate**. The human ear can perceive frequencies between 20-Hz[24] and 20-kHz, while the human voice can typically produce frequencies within the more limited range of 40-Hz to 4-kHz. In order to encode the original analog (i.e. represented by values that vary continuously over time) speech signal into a digital format without loss of fidelity, Nyquist's Theory implies that the signal must be sampled at twice the highest frequency in the sample range. So in order to faithfully represent the human voice in digital values, the signal must be sampled at $2 \times 4\text{-kHz} = 8\text{-kHz}$, the rate assumed by the PSTN to reproduce human conversation.

- **8-bit Quantization**. The number of bits used to store the sampled values of the original analog signal determines the number of levels that can be represented. For example, 8-bits can only represent 256 distinct values, whereas 16-bits can distinguish 65,536 values. Quantization introduces error into the digital signal because a finite set of values can never perfectly represent the infinite set of values assumed by the analog input, so there is a tradeoff between storage cost and the fidelity of the digital audio signal-the more bits, the better the sound but the larger the data storage. Given the limited amplitude range of human speech, 8-bits can produce audio quality suitable for telephone conversation.

- **Mono(phonic) Channel**. Given an appropriate recording environment, a single audio signal can be decomposed into constituent audio signals. For example, the performance of a musical ensemble can be reproduced by playing the recordings of the individual instruments

[24] The frequency of sound waves is measured in Hertz (abbreviated Hz), which is simply cycles per second.

simultaneously; however, the whole is less than its constituent parts, since separate signals can capture level and phase relationships that replicate the directional cues of the original sound source. A monophonic channel represents the audio source as a single audio signal, whereas a stereophonic channel uses typically two signals to represent the source. Since telephone conversations typically involve only single audio sources (i.e. people) on each end of the connection, a "mono" signal is sufficient. In contrast, music recordings (typically comprised of multiple vocal and instrumental sources) are represented by "stereo" systems.

- **μ-law PCM Encoding.** Uniform pulse code modulation (PCM) is an encoding method where the quantizer values are uniformly spaced - essentially an uncompressed audio format. If the spacing is changed from uniform to logarithmic, the range of values can be increased at the expense of nonlinear encoding accuracy: the logarithmic transform focuses on the lower amplitudes while providing sparse coverage on higher amplitudes. Since human conversation normally occurs in the lower amplitude ranges, the increased fidelity at this range easily justifies the loss of fidelity at the higher, infrequently-recurring amplitudes. *μ-law* and *A-law* are two commonly used logarithmic transformations, the former being the North American and Japanese standards, and the latter employed in Europe and other countries.[25]

While audio, including human speech, can be represented in any format in a private network, the public telephone network will only accept audio consistent with the 8-kHz/8-bit/mono/μlaw PCM format.[26] From the perspective of bandwidth requirements, a single call requires 8-kHz × 8-bit = 64-kbps (kilobits per second) of network throughput. This should reinforce the connection between the PSTN call and Internet data capacity: e.g. a T1 can carry 1,544-kbps of data, which is equivalent to 1,544-kbps/T1 ÷ 64-kbps/call = 24 calls/T1.

The issue is not whether the PSTN standard is still capable of supporting telephony service, but whether it provides an appropriate substrate for Voice Web applications. Entertainment and information-focused applications, such as those proffered by voice portals, often seek to repurpose audio content already available on the Internet and from more traditional media sources, neither of which observe the PSTN's audio restrictions. For example, consider a Voice Web business that competes with the traditional radio

[25] The μ-law and A-law PCM encoding methods are formally specified in the International Telecommunication Union - Telecommunication Standardization Sector (ITU-T) Recommendation G.711, *Pulse Code Modulation (PCM) of Voice Frequencies*.

[26] While this paper presents information relevant to the global voice industry, many of the technological and market assumptions are tailored to the North American market, e.g. the assumption of the μ-law encoding format.

market by delivering songs from popular CDs on demand: the caller essentially creates her own radio station by customizing the selection and order of songs. Unlike human conversation, music is typically composed of audio from multiple sources and with a much broader range of frequencies. In order to capture the richness and complexity of performance music, CD manufacturers employ the 44.1-kHz/16-bit/stereo/Red Book format, jointly developed by Philips N.V. and Sony Corporation in 1980.[27] There are a variety of transcoding tools available to convert a Red Book signal into the appropriate PSTN format, but the 8-kHz/8-bit/mono standard exacts a heavy price: dramatically reduced signal fidelity. While an audio engineer's ability to doctor a signal should never be underestimated, the resulting PSTN version of the song will never approximate the quality of an in-dash CD that utilizes the car's speaker system. So the consumer is forced into making a choice between on-demand music with poor sound quality, or a limited number of radio stations with high fidelity.

CDs are not the only example of audio sources that would be appreciably compromised by the PSTN. Traditional radio, television, and cable broadcast networks produce content that is not adequately represented by the PSTN standard: audio signals that fall outside the range of human speech, a richness of sound that would be smeared together by 8-bit quantization, and directional cues lost on a monophonic channel. Even content formatted for web distribution (e.g. MP3 and RealAudio) is targeted to computers with audio adjuncts like stereo speakers and dedicated sound cards. The issue is not whether a transcoding tool exists to bring external audio signals into strict compliance with the 8-kHz/8-bit/mono standard, but whether the obligatory loss of sound quality is justified by the additional utility of the speech-interactive application.

3.1.3. Standardization

While there is only a handful of telephony card vendors that offer the on-board pre-processing functionality (e.g. echo cancellation) desired - though not necessarily required - by speech gateway architects, there are a large number of general-purpose interface vendors. From the perspective of both the developer and the gateway architect, a consistent application programming interface (API) would positively contribute to code portability. Microsoft's Telephony API (TAPI)[28] attempts to provide this consistent abstraction across compliant vendors, but it has met with little success in the speech application community. In practice, the programming languages provided by the ASR vendors (oddly enough)

---

[27] *A Fundamental Introduction to the Compact Disc Player* by Grant Erickson. See http://www.tc.umn.edu/~erick205/Papers/paper.html.

[28] See http://www.microsoft.com/TechNet/winnt/Winntas/prodfact/tapi2.asp.

provide this abstraction for the programmer, and the gateway architect is left to contend with the interface vendor's proprietary API (e.g. Dialogic's CT Media[29]).

3.2. Next-Generation Telephony Interfaces

3.2.1. Packet-Switched Telephony

Despite continued improvements in density and scalability, the circuit-switched interface is approaching the end of its useful product cycle, particularly for large deployments. The trend toward packet telephony is being driven by market forces much larger than the insular interests of the voice industry, and many of the same motivations apply, including the need for exponential improvements in both scalability and density. Apart from pure telephony, most speech applications do not require nor anticipate continuous input from the user: there are variable-length pauses during the points at which the caller is listening to the output, considering her next command, etc. Whereas a circuit-switched port is occupied during these periods of non-productive input (i.e. caller silence), a packet interface effectively services the only the "active" portions of each call on the system. This positive effect of statistical multiplexing affords potentially exponential gains in call density. Other advantages include:

- **Interface Consistency**. Instead of having to manage separate pools of PSTN and Internet interfaces, all communications to and from the speech gateway can conveyed over commodity data interfaces.
- **Flexible Input and Output**. Unlike the telephony-centric PSTN, the Internet provides a generic bit conveyance service with no inherent restrictions on formats or fidelity, apart for the obvious effect of bandwidth scarcity on real-time transmission. For example, these next-generation interfaces can take advantage of algorithms that compress human speech from 64-kbps to under 8-kbps with little effect on fidelity, yielding a significant reduction in the bandwidth requirement for a fixed number of callers. This payload "agnosticism" also anticipates service models that embrace input types beyond speech and output types beyond audio, e.g. Mobile Web may evolve to support simultaneous output of audio and text for browser-enabled cell phones.
- **Cost Savings**. While the packet telephony service provider market has not yet materialized domestically, the theory is that the superior efficiency of packet switching would translate into

[29] See http://www.dialogic.com/products/ctmedia/.

one-time and recurring connectivity savings for the voice service providers that maintain the speech gateways.

The underlying assumption that enables all of these benefits above is that the packet networks over which the incoming speech signals travel are capable of meeting the demands of real-time telephony.

The PSTN, not unlike the Internet, is a logical network comprised of many technologies. Most residential and enterprise users encounter telephony as POTS and T-carrier links, respectively, but there a variety of network technologies that the carriers have deployed within their own networks and for larger customers. For example, Asynchronous Transfer Mode (ATM) is a cell-based[30] technology that carriers employ to carry both data and voice traffic without the T-carrier's inefficient time-division multiplexing (TDM) scheme. The common misconception that packet networks are only recently capable of supporting toll-grade telephony overlooks the long history of packet switching in the carrier network interior: only the traditional "last-mile" elements rely on circuit-switching. However, whereas ATM was designed specifically to support cell-switched multimedia services through QoS provisioning, IP was not. So the current inability of IP-based networks to support QoS across organizational boundaries should not lead the reader to conclude that packet-switched networks cannot support telephony. To the contrary, in addition to its role within the carrier network, ATM or SONET/SDH is routinely employed at inter-carrier exchanges, and large enterprises already have the option of ATM adapters to support traditional call center functions.

The theoretical appeal of extending these carrier packet-telephony technologies to the customer (i.e. VSP) premises must be tempered with the reality that, for reasons outside the immediate scope of this analysis, the ATM-to-the-enterprise movement seems to have stalled before it even started.[31] Meanwhile, the growth of IP-based networks to accommodate the growth of the Internet has spurred academic and corporate research to discover means by which real-time services (notably voice and video) can be conveyed over the non-QoS-aware Internet.[32] For both technical and economic considerations (e.g. how to price service levels across network boundaries), there is no generally available QoS technology

[30] A *cell* is simply a packet of fixed-size.
[31] Parallel to the demand for large-scale VSP services, there was an emerging multi-service trend that uses ATM to dynamically assign the bandwidth of a single DSL pipe to voice, data, video, etc. These ATM-to-the-premises projects were big news in the past few years, but they have amounted to little more than media hype. Examples include: Sprint ION (http://ion.sprint.com), WorldCom On Net (http://www.worldcom.com/for_your_business/on_net/), AT&T INC (http://www.ipservices.att.com/products/).
[32] Refer to the Internet Engineering Task Force's Transport Working Group to sample the various and sundry suggestions for creating a QoS-aware Internet: http://www.ietf.org/html.charters/wg-dir.html#Transport_Area.

deployed throughout the Internet. The absence of supported QoS mechanisms generally leads to a degenerate telephony signal - static, dropped speech – that significantly reduces the accuracy of the speech recognition engine.

3.2.2. Hybrid Circuit/Packet-Switched Architecture

While QoS mechanisms are generally unavailable across multiple packet networks, this does not preclude the application of these QoS technologies within private networks, especially enterprise and service provider intranets. As the name would suggest, a Voice over IP (VoIP) gateway mates the PSTN (typically T1 links) to an intranet (typically Ethernet) and enforces switching policies that ensure sufficient QoS for carrying real-time voice, usually over dedicated network segments unburdened by "bursty" data traffic. In effect, a virtual telephony card interface is distributed across the VoIP and speech gateways.

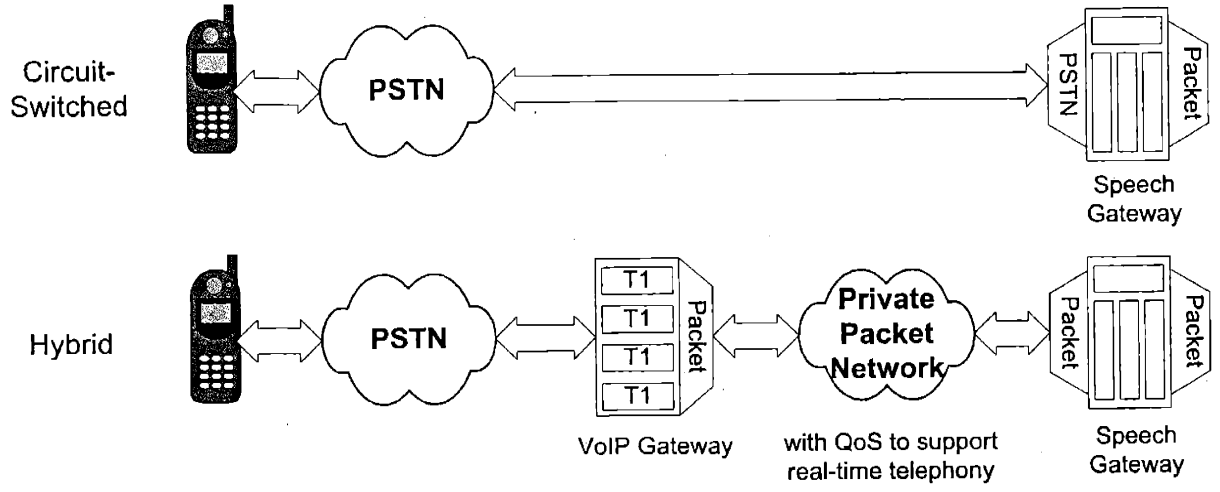

Figure 12. Circuit-Switched vs. Hybrid Circuit/Packet-Switched Telephony Architecture

While the technological viability and superiority of this hybrid architecture are generally accepted, the structural analysis in Part II will revisit this configuration to address the matter of whether operational and financial gains for the VSP are in fact realized.

3.2.3. Packet-Switched Architecture

While the continuing evolution of mobile telephony networks into general data networks creates the potential for "pure" packet telephony directly from the handset to the speech gateway, ensuring QoS end-to-end derails this possibility so long as the packets travel across organizational boundaries. However, there is a unique situation in which true packet telephony does not require the coordination of multiple agencies: the carrier and the VSP are the same organization. Since a single carrier can certainly enforce QoS across its own network, the VoIP gateway is essentially eliminated and the private circuit- and packet-switched resources are combined into a single network. Figure 13 below illustrates this entirely packet-switched architecture, assuming that the network supports QoS levels sufficient for real-time telephony and that the handset is endowed with either software or firmware to perform the customary ASR pre-processing:

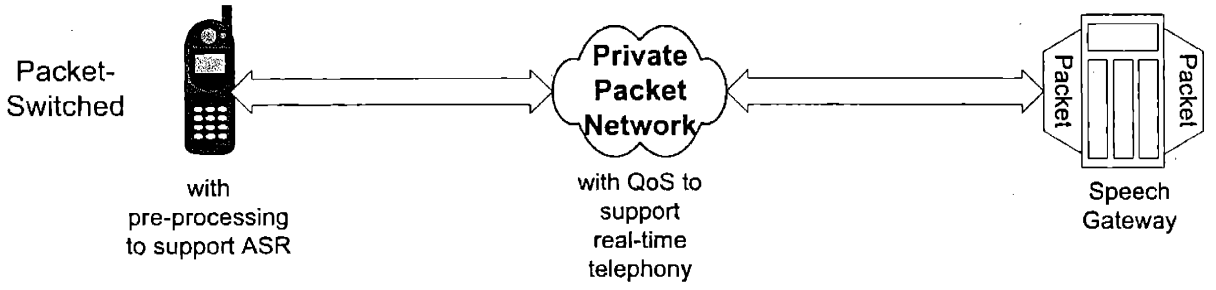

Figure 13. Packet-Switched Telephony Architecture

If and when QoS contracts are enforced across multiple organizations generally, neither the handset nor the speech gateway would require modification, so this unified carrier/VSP architecture anticipates and potentially drives the evolution of the carrier industry. Since this paper is focused on the short-term factors required for the formation of the Voice Web, the reality of the circuit-switched PSTN is accepted as the reference architecture; however, the subsequent strategic analysis will examine the role of handset manufacturers and carriers in the long-term development of the voice industry.

3.3. Data Interface

While the conventional SIT-oriented telephony interface is an expensive device available from only a few vendors, the data interface is often nothing more complex than a commodity Ethernet card, easily

purchased for less than $100[33]. A Fast Ethernet segment is theoretically capable of 100-Mbps, capable of sustaining over 1,562 simultaneous telephone-quality conversations. Like the telephone interface, multiple data interfaces can be pooled together to form a virtual interface that can be subtended by routers and connected to the Internet.

3.4. Summary

While traditional telephony interfaces have supported SIT applications for many years, their high per-port costs and their inflexible, inefficient apportionment among concurrent callers collectively question their appropriateness in distributing a mass media product to hundreds of thousands. In addition, the rigid audio format of the current 64-kbps/call PSTN infrastructure precludes high-fidelity audio content in any application. The commodity data interface, by contrast, features an efficient access mechanism that offers a low marginal cost that can flexibly support a variety of input and output formats. In recognition of this cost and engineering efficiency, the next-generation interface will attach to telephony-grade internets and effectively evolve into just another data interface – provided that a general market for private, QoS-aware networks develops concurrently. It should also be noted that while flexible support for input and output formats is an exciting prospect, it would require similar evolution of the end user device industry and negate the market advantage of the nearly ubiquitous, albeit low-fidelity, telephone.

[33] The 10BaseT (10-Mbps) Ethernet standard is defined by the Institute for Electrical and Electronic Engineers (IEEE) as IEEE Standard 802.3, and the 100BaseT (100-Mbps) Fast Ethernet standard is described in IEEE 802.3u.

Chapter 4. INPUT TECHNOLOGY - AUTOMATED SPEECH RECOGNITION

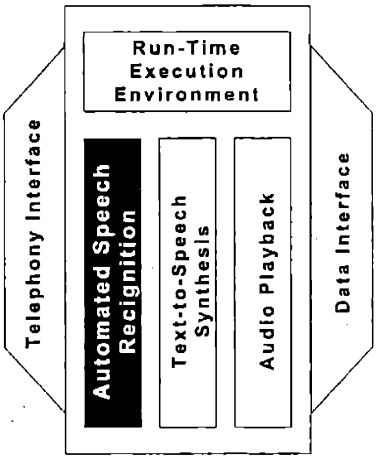

Figure 14. Speech Gateway (Currently on Input Technology)

The goal of ASR technology is to accurately and consistently recognize the intention behind a user's speech input, and then signal that intention to the application for further processing. Given the obvious variability and scope of spoken languages, the task of recognition must necessarily be circumscribed in order to achieve acceptable error rates.

4.1. Basics of Speech Recognition

4.1.1. Traditional ASR Architecture

The art of recognizing spoken input is ultimately rooted in matching the incoming signal (in this case, a speech fragment delivered over the telephone system) against a set of reference patterns or pattern rules.[34] At the core of all modern ASR systems are complex statistical models that are able to characterize the properties of the sounds of a given language. Just as the basic building blocks of the written language are the letters of the alphabet, the atomic elements of the spoken language are known as *phonemes*. Humans string phonemes together to form single words, and in turn arrange these words in a linear composition,

[34] For the purposes of this paper, the term "ASR" refers to both speech and DTMF recognition. Since the recognition process simply matches the incoming signal against a reference set of signals, the always-consistent DTMF "tones" are the easiest inputs to recognize in practice.

making sure to avoid offending the rules of grammar. ASR systems essentially extract these phonemes from a speech signal for comparison against a reference set of known phonetic sequences. Figure 15 below outlines the basic components of the speech recognition process:

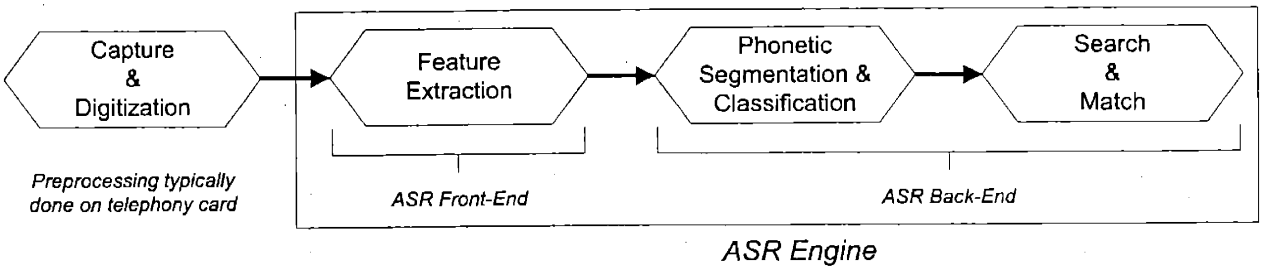

Figure 15. Basic Components of Automated Speech Recognition

1. **Capture & Digitization**. Already discussed in Chapter 3 is the role of the telephony interface as the first piece of the ASR process. In addition to the rudimentary format conversion of the incoming telephony signal (no digitization step likely considering that most incoming links are already digital, e.g. T-carrier), the telephony card is tasked with ASR-specific functions (e.g. endpointing, energy detection) that collectively determine whether a valid speech event has occurred and requires further processing. While these preprocessing tasks can be performed in software or disregarded altogether, they are almost universally employed to increase the efficiency of the ASR resource pool.

2. **ASR Front-End: Feature Extraction**. Before the individual phonemes can be identified, the captured signal must be converted into a form more suitable for distinguishing these atomic speech sounds. After applying basic filters (e.g. offset compensation) to address the variability of the input, the speech signal is partitioned into frames and mapped to the frequency domain to isolate those portions most relevant to human speech (e.g. low frequency portions de-emphasized). Using this spectral representation as the basis, a set of *parameters* (or *features*) is calculated such that each frame is compactly represented by these indicative parameters, rather than having to reproduce the entire speech signal. As will be discussed later, the economy of this compact parameter form or *feature vector* creates the possibility for distributed speech recognition.

3. **ASR Back-End**:

a. **Phonetic Segmentation & Classification**. Once the feature vector is generated by the ASR Front-End, it is decomposed into individual speech sounds, which are then matched against a reference set of phonemes appropriate for the specific language. A match is not a binary decision but rather a score of statistical proximity between the segment of the multidimensional feature vector and the reference phoneme under consideration. Note that this stage may produce multiple probable segmentations of the same speech fragment, all of which are inherited by the next phase.

b. **Search & Match**. Given this network of possible phonetic segmentations and probabilities, this final stage compares these phoneme sequences against the reference set appropriate for the given user dialog. For example, if the valid or active search space is every individual word in the American English language, then the ASR engine is armed with an English-to-phonetic dictionary against which to look up phoneme sequences. As with the segmentation and classification process, phonetic "matches" are statistically along many parameters, permitting flexibility in how results are reported to the application. For example, the engine can return the entire array of match values for each reference pattern, the arrays can be reduced to single numbers of heuristic significance (e.g. match "score" on a scale from 0 to 1), or a default selection process can simply signal to the program the most likely match (if any). Most ASR vendors provide multiple access points for these results to afford the most flexibility to the programmer. "Default" matching (heuristically best match is reported) and "N-best" matching (the N-best matches are reported to the program layer, where N is programmer-defined) are the most common formats for results.

An example will best illustrate the concepts of segmentation and matching to produce an ASR result. Suppose that the caller has just uttered the phrase "I enjoy the performance." The telephony card, which had previously been monitoring the call through long periods of silence and background noise, now detects the beginning of a valid speech signal. Once the endpointing algorithm is satisfied that the caller has completed the phrase, the fragment is digitized (if not already) and forwarded to the ASR front-end. After the salient feature parameters have been extracted from the signal and it is rendered in the more compact spectral form, it is forwarded to the back-end ASR engine for phonetic classification. If the active search space is the entire English lexicon, then the ASR engine essentially performs a reverse-lookup in its internal English-to-phoneme dictionary, e.g. \in-'joi\ maps to ENJOY and \th\ maps to THE.[35]

[35] The convention to describe ASR vocabulary elements is as follows: the digitized speech fragment in both temporal and spectral forms is represented by the intended text within angles (e.g. <hello>), the phonetic

The application that is handling this interaction is returned the complete text string I ENJOY THE PERFORMANCE, likely accompanied with a heuristic score of confidence. Figure 16 summarizes this transformation from spoken audio fragment to text.

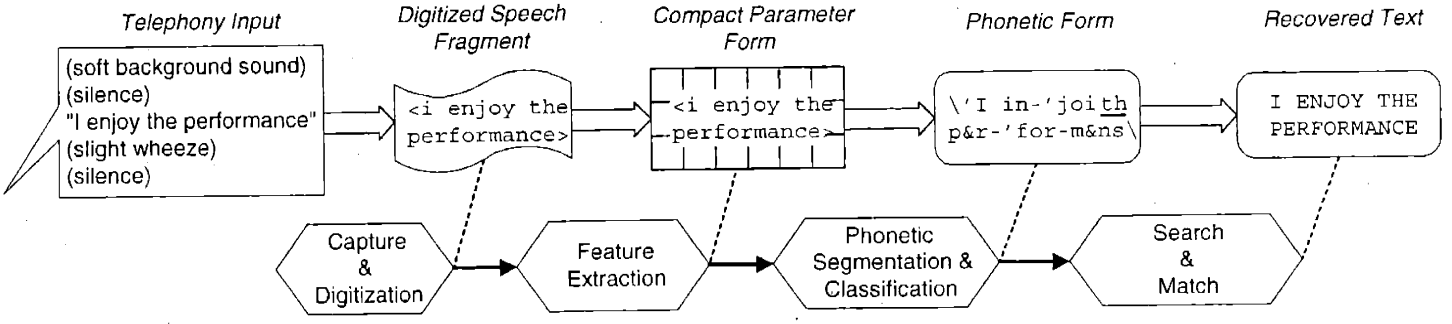

Figure 16. Example of the ASR Process

Combining the logistics of ASR processing in Figure 16 above with the Reference Model for Speech-Interactive Telephony depicted in Figure 9 (page 32) yields the following architectural model and observations:

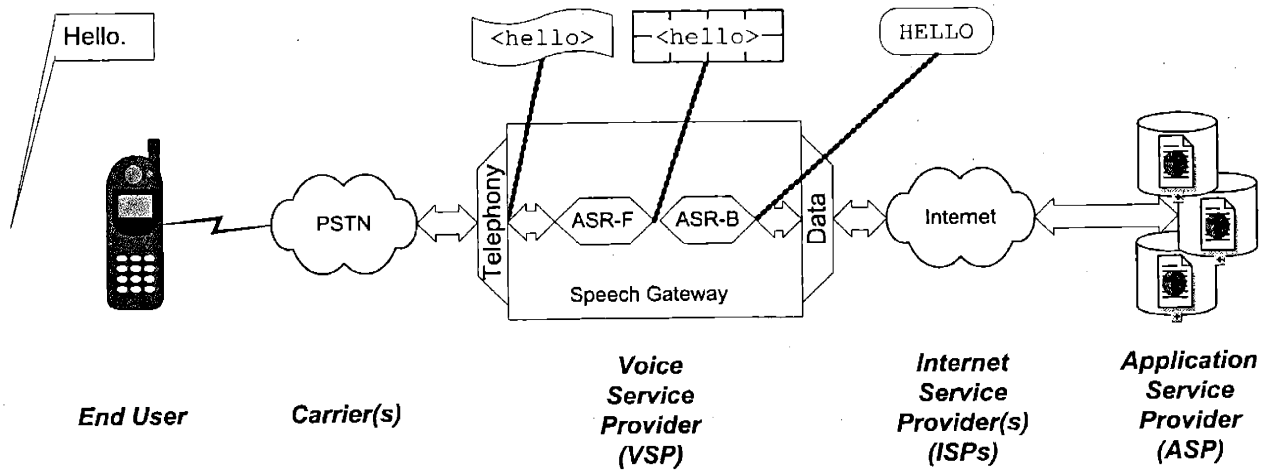

Figure 17. Traditional ASR Architecture

- Both the ASR Front-End (ASR-F) and ASR Back-End (ASR-B) are integrated into the speech gateway in a traditional ASR architecture. The carrier network simply conveys the audio signal to the speech gateway, which then assumes control of the process until ultimately a text string is

---

decomposition of the raw speech signal is represented by the Merriam-Webster pronunciation within reversed virgules (e.g. \h&-'lO\), and the text mappings themselves are represented in small caps (e.g. HELLO).

recovered. Note that the phonetic form of the speech input does not appear in this illustration because it is an internal result of the ASR-B process.

- The PSTN is considered to be a single network, but it should be remembered that it is comprised of multiple private telephone networks: the end user and VSP are free to choose different telephony providers, hence the plural form of "carriers" in Figure 17 above. The same reasoning motivates the presence of multiple ISPs since the VSP and ASP don't necessarily share the same ISP.

- The digitization function performed by the telephony board does not imply that the communications path from the handset to the speech gateway is not digital, particularly since most mobile networks employ some variant of digital access. Since each carrier may implement telephony over a variety of technologies (e.g. T1, VoIP, ATM), the telephony interface's re-sampling and pre-processing normalizes the input signals and abstracts away the interface complexity from the ASR engine.

4.2. Speech Recognition Grammars

Apart from general improvements in ASR algorithms and computer performance, the speech application programmer can dramatically enhance both the accuracy and efficiency by supplying some information in addition to the raw speech signal: a *grammar specification.* A grammar defines the search space for the matching process, and it can be expressed simply as a list of words and phrases, or as a complex set of rules that specify not only vocabulary but also permissible relationships between the words and phrases.[36] In the example above, the entire English lexicon serves as the grammar for the speech recognition problem: each phoneme segment is compared against potentially every word in the English language for a possible match. However, the search space can dramatically reduced if the programmer can design the application to limit the range of valid inputs. For example, instead of assuming that any English input is valid, suppose that the application is much more limited and accepts only two input phrases as valid: "I enjoy the performance" and "I adore the theater."

```
Grammar =    {   ADORE = \&-'dOr\
```

[36] One common way of specifying these grammar rules, and the convention employed throughout this paper, is to use *regular expressions.*

```
    ENJOY = \in-'joi\
    I = \'I\
    PERFORMANCE = \p&r-'for-m&ns\
    THE = \th\
    THEATER = \'thE-&-t&r\
}
```

The specification of a more limited grammar does not alter the basic processes of speech recognition but simply limits the search space for potential matches. By reducing the number of potential matches from almost 1 million (not including industry-specific or scientific jargon) to only 6, the search time is dramatically decreased, resulting in either higher recognition throughput or a higher caller to ASR engine utilization ratio. A second benefit of this reduced grammar is increased accuracy. For example, "I" and "eye" are homonyms, and only an intimate understanding of English grammar would allow an ASR engine to differentiate between the two. But since "I" alone appears in the grammar above, the ASR engine can quickly make the proper match without any understanding of homonyms or grammar. Another example would be the difficulty of distinguishing between "a door" and "adore": the two-word phrase is typically uttered quickly enough to frustrate attempts to isolate the brief pause between the words. Again, there is no confusion since only "adore" appears in the sample grammar. Examples like these are common in the English language, so a well-designed grammar can reduce this ambiguity by implicitly eliminating other high-scoring matches.

Since grammars allow the specification of arbitrary phoneme sequences, each grammar element need not be limited to single words. To continue the example above, the grammar is reduced to only 2 elements:

```
Grammar = {   I ENJOY THE PERFORMANCE = \'I in-'joi th p&r-'for-m&ns\
              I ADORE THE THEATER = \'I &-'dOr th 'thE-&-t&r\
          }
```

Instead of burdening the recognizer from having to piece together the speech fragment from individual elements within the grammar, the programmer may opt to have the complete fragment compared against the target phrase. Of course, this removes previously allowable variations such as "I enjoy performance" and even " enjoy the theater," but the programmer and the application designer need to make that decision. There is a delicate balance between grammar elements so short that they sound very similar (e.g. "our" and "are"), and elements so long that a single mistake or variation will cause a misrecognition (e.g.

"I enjoyed the performance" may be rejected). These grammar decisions are guided largely by the application itself: a dictation program would want to accurately capture free-form input, whereas an stock quote program would want to give the caller a finite set of verbal commands to use. A speech application programmer must always appreciate the relationship between flexibility and accuracy.

In addition to performance and accuracy, there is a third reason for specifying a more limited grammar: interpretation. While the ASR engine can recover a text string from a speech fragment, it cannot infer the meaning of that text sequence. A grammar not only guides the engine through the most efficient path of searching and matching, but it also effectively defines the set of all possible meanings. An unrestricted (i.e. grammar specification is the entire English language) ASR engine could potentially report almost any phrase to the application, but it would be impossible to design a program to interpret and act on these millions of natural language combinations. A programmer constructs a grammar to define all the possible directives that the application is willing or able to interpret at each point in the program flow, and the caller has to follow the convention of the application in order to be understood. For example, the trivial grammar {DELETE THE FILE = di-'lEt th 'fIl} not only implies that the program knows how to interpret this action, but also that it cannot and will not act on a random utterance like "shoot the dog." In effect, the grammar becomes part of the user interface and the program logic by defining the set of commands at each interaction. The challenge is to train the caller to restrict his speech to this finite set of directives while maintaining an aesthetically pleasing user interface.

4.3. Airport Application Example[37]

The running example of an information number for an airport will demonstrate the grammar concepts above. Logan International Airport receives hundreds of calls each day regarding flight arrivals and departures, weather, parking information, etc. Distributing this information is an integral part of the airport's responsibility, but it is costly to hire live operators to staff a call center, especially considering that most of the information is either read from a script or easily accessed from a database. In order to cut costs and to provide 24-hour support, Logan decides to build a SIT application-one that will rely heavily upon grammars to support the relevant functions.

[37] Disclaimer: The airport example is designed to illustrate the challenges of designing a speech-interactive application with grammars, and not as a textbook example of good human factors design.

4.3.1. Yes or No

The application developers are concerned that callers might be initially uncomfortable using the automated system, so they would like to allow the user to select whether they want to use the new system or simply get transferred to a live operator. So when a client calls the number, she is greeted with the following prompt: "Welcome to the Boston Logan International Airport's Information Line. Please say 'yes' if you would like to use our automated system for flight arrivals and departures, or 'no' if you would prefer to speak with an operator." A simple grammar will restrict the search space from the entire English language to only two single-word elements:

```
Grammar =    {   YES = \'yes\
                 NO = \'nO\
             }
```

This small grammar has not only reduced the computational complexity to support this step of the application by many orders of magnitude, but the accuracy is improved as well, e.g. there is no confusion between "no" and "know" since the latter is not a valid input.

After initial testing of this initial prompt with live callers, it is discovered that a significant percentage of speech inputs are being rejected because no match is found. The callers are indeed responding to the question, but some of them are using alternate or colloquial pronunciations. For example, in addition to the canonical form, clients might pronounce the word as \'yas\, or they might revert to something more familiar like "yeah" (pronounced \'ye-&\) . The programmer elects to simply add elements to the existing grammar to capture these acceptable variances.

```
Grammar =    {   YES = \'yes\, \'yas\, \'ye-&\, \'y&p\
                 NO = \'nO\, \'na\, \'nOp\
             }
```

4.3.2. Airline Name

If the caller opts to try the automated system, she is greeted with a new prompt: "Please clearly speak the name of the airline for which you want arrival and departure information." Instead of a simple yes/no

grammar, the programmer must create a new grammar that specifies the name of every airline at Logan Airport. As in the previous prompt, the grammar must account for alternate (e.g. 'air' can be \'ar\ or \'er\) or colloquial (e.g. 'US Airways' is often shortened to 'US Air') pronunciations:

```
Grammar = {   AIR CANADA = \'ar 'ka-n&-d&\, \'er 'ka-n&-d&\
              AIR FRANCE = \'ar 'frants\, \'ar 'frans\, \'er 'frants\,
              \'er 'frans\,
              . . .
              US AIRWAYS = \'yü 'es 'ar-"wAz\, \'yü 'es 'er-"wAz\,
              \'yü 'es 'ar\, \'yü 'es 'er\,
              VIRGIN ATLANTIC = \'v&r-j&n &t-'lan-tik\, \'v&r-j&n at-
              'lan-tik\, \'v&r-j&n\
          }
```

Even though there are only about 30 airlines that fly through Logan Airport, it's apparent that the grammar specification to support this stage of the program could be quite large. Contrast this design task with that of a similar application that is delivered via the HTML Web. The user is not asked to key in the name of the airline manually, but is typically asked to select from a pre-determined drop-down list of valid airline names. So even if the user would say "US Air" in conversation, in her mind she knows that "US Airways" is an equivalent input and would easily make the right selection from the menu. Imagine how much more challenging it would be if a web site designer had only free-form text input for all fields (e.g. if a state name is misspelled, are the closest matches offered?). Even though the pre-recorded prompts can exhaustively enumerate the options and their canonical pronunciations (though in this case of airlines names this would be a prohibitively long prompt), the conversational expectation created by the phone interface opens the proverbial Pandora's box of variants.

The essential difference between speech and graphical user interfaces is that there is no way to restrict speech input. While the grammar specification is similar to the drop-down menu in that all the possible options that the program is able to accept are enumerated, the grammar does not constrain the user's speech but rather constrains only the recognition task. Graphical interfaces that concretely limit the input through menus not only constrain the response but also the input as well. So not only must the grammar be expertly designed and exhaustively tested to capture all valid inputs, but the application itself must extensively cue or educate the caller to *self-restrict* her speech to ensure proper operation.

4.3.3. Flight Number

Once the airline has been successfully selected, the application will issue the following prompt: "Please clearly speak the number of the flight for the airline you selected." Since flight numbers are natural numbers between 1 and 9999, there are at first glance almost 10,000 spoken numbers that need to be included in the grammar, but the problem is much more complex than simply enumerating each possible combination as a separate element in the grammar specification. Apart from nuances in pronunciation, there are aggregate variations in how each number is lexically rendered. For example, the flight number "0203" can be spelled out as "two-o-three," "two-zero-three," "zero-two-zero-three," "o-two-o-three," etc. An exhaustive enumeration would be prohibitive and would contribute to human errors of omission, so it should be remembered that grammar specifications do not simply enumerate vocabulary elements but also the relationships between them - this is how a dictation program is able to understand both individual words and grammar rules. While each vendor may implement this feature differently, they often employ some form of the *regular expression*:

```
Grammar = {   NUMBER = NUMBER^DIGIT
              DIGIT = ZERO,ONE, TWO, ... ,NINE
              ZERO = \'zE-rO\, \'zir-O\, \'O\
              ONE = \'w&n\
              TWO = \'tü\
              . . .
              NINE = \'nIn\
          }
```

The syntax of the regular expression is not important, but the concept that grammars can be flexibly built using relationships and not just brute-force enumeration is key to solving nontrivial speech recognition problems. Of course, the caller may not choose to sound out each digit and instead elect more "formal" forms (e.g. "two-hundred-and-three," "two-hundred-three"), but these variations can also be captured in a series of regular expressions, albeit more complicated ones.

Rather than burden the programmer with the task of developing the robust grammars for these and other commonly recurring requests (e.g. credit card number, dates, times), ASR vendors typically provide a library of higher-level objects that supply the appropriate grammars and logic to the engine. For example, the programmer would simply invoke the function *getNaturalNumber(Range=[0,9999])* in order to

retrieve the flight number, abstracting away all the design and testing complexity. As will be explored in later sections, these *reusable dialog components* (RDCs) utilize the ASR vendor's specific grammar syntax and programming conventions, so they are not reusable across ASR platforms.

4.3.4. Challenge of Developing Speech Applications

So for this simple airport application, each request for information from the user involves matching the recovered phoneme sequence against a particular grammar specification. Grammars increase the efficiency and accuracy by restricting the search space, and form the set of operations that the application is capable of interpreting. Because of the extensive variety of canonical and colloquial pronunciations, grammars must be expertly designed and thoroughly tested, prompting the creation of programming objects that support the most common dialogs.

Much more than a curiosity for application programmers, grammars will exert great influence over whether the vision of Voice Web is achievable. An important factor in the Voice Web's success is the extent to which the technology is accessible and understandable to the Web community. With the aid of software tools and perhaps a good manual, a person can create a fully functional HTML website with very little effort or knowledge beyond simple layout; by contrast, a person building a speech website would require knowledge about constructing grammars. While tools exist to create grammars for commonly requested functions, the user is largely on her own if she wants to create, for example, a voice-index to her online music collection. It remains to be seen whether the dissemination of this knowledge and expertise is simply a matter of time: the formative text editor "era" of the HTML Web stands in stark contrast to the graphical editor and middleware market that currently exists. On the other hand, perhaps the speech application watermark is fundamentally too high to foster a broad community of Voice Web development. The level of participation on online Voice Web development communities is encouraging, but it remains to be seen whether the resulting applications have successfully met the challenges of dialog and grammar design.

4.4. Advanced ASR Functionalities

Basic speech recognition conditioned with grammar specifications forms the foundation for virtually all SIT applications, but many ASR vendors extend this core feature set to improve performance, attend to user interface concerns, and support dynamic application content and logic.

4.4.1. Pre-Compiled vs. Runtime Compilation

In the case of the airport application, the grammars are "static" in that they do not vary based on caller's identity or previous responses: there is no runtime logic that tailors the grammars. Just as "C" program cannot be executed until it is compiled into a form suitable for machine execution, the grammar must also be compiled into a format suitable for the ASR engine. The standard approach is to pre-compile the grammar files and load them onto the speech gateway for ready access by the application.

However, if the airport application supported registration of users and customizing the airline options to list only "favorite" airlines, the task of recognition could be greatly improved. The user input is no longer compared against a lengthy cadre of airlines but rather against a very small and specific list. The computational gain in speed and accuracy, however, must be offset by the computational loss in compiling grammars in real-time. Since each user has unique grammar for a certain subset of dialogs, it potentially will not be created until she identifies herself to the system. The application code would dynamically create an appropriate grammar based on her specific preferences, which would then need to be compiled immediately before recognition could occur. Since the Voice Web purports to support heterogeneous set of applications, each with varying levels of user customization, real-time grammar compilation is an absolute necessity.

4.4.2. Word Spotting

Leaving aside the promise of natural language input, most speech applications focus on improving recognition by keeping the task as simple as possible. Rather than encouraging free-form input from the user, a well-designed application would train users to speak within the confines of a constrained vocabulary. While barking terse commands to a machine is hardly redolent of natural conversation, it saves the programmer and ASR engine from having to actually understand the meaning of more verbose directives.

Despite the best efforts of the application designer, callers will invariably revert to a more loose construction of commands. While they may not form full sentences, it would not be unexpected to encounter inventive caller variations. For example, a portal application that supports a "go sports" command may routinely encounter in practice: "go to sports," "play sports," "yeah, sports," "um, sports,"

"sports" etc. These variations could be added to the grammar to catch these as valid inputs, but the permutation and novelty of the spoken word makes such a process intractable, e.g. "um, yeah, go play sports." *Word spotting* is the ability to spot a pattern within a larger input pattern allows the ASR engine to interpret a verbose input by matching only a fragment to the specified grammar. So the phrase "um, yeah, go play sports" could be interpreted as "sports" simply because of the appearance of the word. It doesn't matter what all the other words mean or their relationships to the keyword: if "sports" appears and no other keywords are triggered, then that's all the application needs to know.

While enabling a pseudo-natural language input, word spotting introduces potential complications. For example, if the caller speaks "I want sports news," and "news" and "sports" are both valid portal keywords, the appropriate course of action is unclear. Encouraging the user to be less specific in their directives potentially creates the false expectation that the ASR engine understands more than it actually does. Structure, while inherently limiting, may very well increase user satisfaction (the ultimate metric) over the long term. Most SIT application developers strike a compromise: concerted training and prompting efforts coupled with word spotting on a limited grammar.

4.4.3. Grammar Scoping

Implicit within the ASR discussion above is that grammars are not active all the time. For example, the name grammar for the airport application is only relevant (hence, active) when the airline selection dialog is being posed. Concurrent with the linear, temporal flow of the application logic is a sequence of grammars to condition and thereby improve the recognition process. In other words, the *scope* or active region of each grammar is normally tied to the dialog being presented to the caller.

In many cases, such as within a multi-tiered application like a voice portal, the application developer may desire to activate certain grammars over an entire set of dialogs. The programmer would prefer to avoid the laborious hand coding and record-keeping required to add that particular grammar to each dialog singly, so a mechanism is needed to flexibly set and manage the grammar's scope. This functionality is similar to the implicit variable scoping of most modern computer languages: each variable defined by the programmer is active only within a certain scope: global, active within a function call only, etc.

Continuing the airport application above, consider the addition of a 3-word grammar that should be active at every point in the call flow: "help," "menu," and "favorites." Rather than arduously adding these

elements to each dialog in succession, the programmer simply anoints this grammar with universal scope and the execution environment handles the bookkeeping of activating and deactivating the grammar when appropriate. It should be noted that scoping is not a binary designation of local versus universal application scope, since many different intermediate scoping levels can be constructed, e.g. the keyword "boxing" may be active at any point within the "sports" section of a voice portal but deactivated in other categories such as "weather," "finance," etc. Many grammars with various levels of scoping can be active at any point in the application flow.

Apart from facilitating the development of single application, the importance of scoping is amplified by any model (including the Voice Web) that supports "nesting" of applications from multiple sources. For example, consider the case of a consumer voice portal that links the category keyword "news" to CNN's own voice application. The portal application will certainly feature a universal command set that should be active within CNN's code; however, CNN certainly has little desire to customize its own application to suit the varied and random interests of every voice portal that refers to it. The mechanism for designating the portal's universal grammars must be able to enforce that scope across organizational boundaries to ensure the seamless integration of these two applications.

4.4.4. Barge-In

Two-party communication, between humans or machines, can be represented stylistically as an alternating, ping-pong-like process in which only person can speak at any given moment; however, reality would suggest that there are legitimate reasons for interruptions. *Barge-in* is the ability of the caller to issue another command while the system is currently playing an audio file, whether it is a prompt for input or a response to a previous directive. For example, barge-in permits the caller to say the name of the airline before the system completes playback of the verbose prompt.

Barge-in assumes particular importance in the context of long-play audio. Suppose that voice portal caller said "CNN" (Cable News Network) which the ASR engine mistook as "TNN" (The Nashville Network), launching the user into a 30-minute continuous program filled with classic nuggets from the Dukes of Hazzard, the pro-am bull riding circuit, and the monster truck schedule. It would be reasonable to assume that once the caller had realized that a mistake had been made, he would like to interrupt the playback in order to re-issue the directive or administer another command.

Apart from supporting user-initiated interruptions of the call flow, barge-in can also simplify the development process by allowing a single design to cater to both the novice and expert users. A verbose, pedantic system that guides the user through each excruciating step of the application flow may indeed be useful for novices, but would grate upon the goodwill of those who are familiar with the application and would like to get to their "destination" as quickly as possible. Barge-in allows an application to be designed for new users while enabling expert users to glide deftly through the menus and options.

Barge-in is an essential feature of any speech gateway; however, its benefits can be offset by inappropriate interruptions of the audio playback. A blaring car horn, an inadvertent cough, or a casual comment from a nearby friend may trigger a speech event, causing the system to interrupt playback and respond with the prompt: "I'm sorry, I did not understand you. Please repeat the command or just say 'help.'" If the external triggers persist, the audio playback will be interrupted continuously, rendering the system unusable. While the telephony card superficially screens the incoming signal to detect valid speech events, a sound of sufficient energy (valid or not) will invariably be passed onto the ASR engine for further examination. Since the mobile environment is typically saturated with random and non-random noise, a robust speech gateway must offer a barge-in facility beyond the simple *energy-based barge-in* initiated directly by the telephony interface. *Recognition-based barge-in* overcomes this limitation by forwarding the alleged speech event to the ASR engine without interruption of the audio stream. If the event is determined to be invalid, the playback will continue and the caller is spared the intrusion.

But the application of the dressing may cause another wound: if a user incorrectly expects the keyword "help" to be active during a long audio playback, the system will ignore the caller's repeated pleas for help and continue the playback uninterrupted. The incorporation of recognition-based barge-in must be tempered with the need to enable a delightful user experience. An example of how recognition-based barge-in can potentially create a frustrating user experience is the *Wildfire*,[38] a speech-interactive personal assistant that employs a female persona of the same name. Wildfire responds to requests to check voice mail, dial a name from a directory, forward a call, incoming call alert, etc. Since the creators envisioned the need to access this virtual assistant during a call, the user can speak the word "Wildfire" at any time to bring the assistant back into command mode. The challenge is to tune the sensitivity of the engine to avoid inadvertent interruptions during phone calls while consistently responding to legitimate requests for assistance. During the author's brief encounter with the Wildfire system, it would take several attempts to

[38] For more information, see http://www.wildfire.com.

get the system to respond during a call, an exercise exacerbated by the audible annoyance in the author's voice after the third consecutive attempt.

Whichever type of barge-in schema is employed, the flexibility gain for the user must be offset against the increased hardware and software costs. The ASR resource pool must be enlarged commensurate with the increased speech event load to maintain the same level of system responsiveness (i.e. throughput).

4.4.5. Modeling for Noise

Most ASR engines were designed to support enterprise-focused applications, so speech recognition occurred in a rather ideal environment: the clarity of landline phones and the relative quiet of the indoors. However, if the author's assumption is correct that the Voice Web's natural setting is the mobile environment, then the ASR engine must be capable of dealing with the vagaries of the cellular network and ambient noise.

Fundamental advances in cellular multiplexing and compression play a decreasing role in signal distortion, but physical factors (e.g. spotty coverage, rain fade, large obstructions) continue to contribute to dropped or garbled calls. Another factor is the cellular handset's general lack of discretion: valid speech and ambient noise are overlaid and sent along as a single audio signal. Noise sources both acute (e.g. person sneezing) and chronic (e.g. constant din of car engine) complicate the already difficult task of speech recognition by reducing the fidelity of the incoming audio signal. People generally perform better than the best ASR engine, yet it is not uncommon to observe people yelling into their handsets and repeating themselves because of poor sound quality, so the challenge to ASR vendors is quite formidable.

Part of the problem is that most ASR engines have been "trained" on spoken data that assumes almost ideal ambient conditions. Spoken language *corpora* - libraries of spoken words from a variety of genders, regions - are used to continuously update and refine the statistical parameters of the ASR engines, so several vendors have turned to corpora that incorporate actual cellular conversations to essentially "retrain" their recognizers. The introduction of "mobile-hardened" ASR engines from both mainstream and boutique vendors signals a growing appreciation for the challenge and opportunity of the mobile speech application space.

4.4.6. Speaker-Independence vs. Speaker-Dependence

In order to reach the widest possible audience, the ASR engine must successfully negotiate human speech of varying dialect, timbre, pitch, etc. The extensive training on spoken language corpora forms the foundation for successful recognition, but several vendors offer the further option of training the engine on the particular user's speech. Consumer dictation programs require users to recite a cadre of linguistically "indicative" words and phrases to refine the engine's parameters, yielding appreciable increases in recognition accuracy. The resulting *speaker-dependent* ASR engine is suitable only for that particular user – a common cold or fatigue can potentially render the system unsuitable even for the target user.

The largely positive effect of speaker dependence must be weighed against the needs of a mass market voice application. Unlike a dictation program that the user will engage repeatedly, a voice portal or Voice Web application encourages a more casual relationship: the consumer has little incentive to train each speech gateway she encounters, especially if she has only a cursory interest in hosted application. Consider the reduction in traffic to HTML Web site if the consumer were expected to complete a 10-page form before seeing even the first page of content. If one of the bedrock principles of the Voice Web is the removal of barriers between destination sites, then the consumer cannot reasonably be expected to engage in a multitude of training sessions.

While it has been suggested that ASR training data be standardized and accessible to all Voice Web destination sites – a sort of speech "wallet" or "profile" - this presupposes architectural similarities between engines that simply don't exist, not to mention an unprecedented level of cooperation between ASR vendors. In the absence of user specific training, *speaker-independent* ASR systems must rely on extensive training on spoken language corpora and grammar-directed pattern matching to support the Voice Web model.

4.4.7. Multiple Languages

While the North American consciousness has only recently awakened to mobile telephony, cellular phones have already become an integral part of the Latin American, European, and Asian societies. These cultures may be longsuffering in their accommodation of the English language, but they will not be impressed by a Voice Web that demands the adoption of an American English accent. Since speaker-interaction lies at the core of the Voice Web, it stands to reason that the ASR technology must be

accomplished enough to support multiple languages. A speech gateway equipment vendor desires the broadest possible market for its platform, but at the same time recognizes that the task of integrating a new ASR engine is nontrivial and can take several months; consequently, many gateway vendors look to ASR suppliers that strike the right balance between superior performance and language coverage.

4.4.8. Speaker Verification

Voice applications rely on the identification of individual customers to support billing, personalization, ad targeting, security access, etc. If security were simply a challenge-response on numeric logins and passwords, a digit string grammar and a database lookup is all that is necessary - the caller could even resort to DTMF entry if she felt uncomfortable speaking the access codes out loud.

Speaker verification (a.k.a. voice recognition, speaker authentication) affords an extra measure of security by matching a speech fragment (e.g. login) against the individual caller's pre-recorded pattern. It can be considered as a limited application of speaker-dependent ASR technology since both the correct identification codes and the proper pronunciation are required to be authenticated. So even if someone were to discover another caller's private access codes, that person would also have to sound identical to the authorized person to gain access.

Speaker verification technology enjoys success in certain niche markets (e.g. identification and monitoring of inmates calling from jail to circumvent harassment and fraud), but it suffers from the same technological and marketing disadvantages as speaker dependent ASR: consumer aversion to training, storage of speech profile, potential consumer frustration when it fails to permit legitimate access, etc.

4.5. Standardization

The speech gateway architect is ultimately concerned with the effort required to integrate a particular ASR engine into the product offering. The commercial ASR industry is competitive, so the gateway designer must select the subset of engines that best addresses the needs of the target market, e.g. a gateway vendor looking to expand into the European market would prefer a single ASR vendor that provides multi-lingual coverage. This integration effort would be greatly furthered by the introduction of a consistent interface to all ASR engines, regardless of vendor. Microsoft's Speech API (SAPI) affords a level of standardization that has been adopted by the leading ASR vendors, but the integration task is still

extensive.[39] Consider the case of an automobile: having a consistent "interface" between the gasoline tank and the axle does not imply that engines are interchangeable. In the same manner, ASR engines with the moniker of "SAPI-compliant" are not commodity elements that can be flexibly interchanged. In practice, most speech gateway vendors select only a few ASR engines given the costs of integration and support.

From the perspective of the application developer, the programming language used to implement the speech application must provide an interface to the ASR engine to access the text recovered from the spoken input – this is directly analogous to providing an API to receive input from the keyboard or mouse in a desktop computer application. Since speech input is rather specialized and requires more programmer direction than other input modes, the ASR vendors have traditionally provided both the API and tools to develop speech applications. Since the dialog language engages not only the ASR engine but the other elements of the speech gateway as well, it provides an integrative abstraction for the entire Voice Web architecture, so the discussion of dialog language standardization will be deferred to the Execution Environment chapter.

Another standard of particular relevance to the programmer is the syntax of the grammar specification language. Given its tight integration with and control over the ASR engine, the syntax and even functionality of grammar languages varies significantly across vendors. Since most speech applications have traditionally been developed for a particular vendor's speech gateway, there was little call for the creation of a common speech grammar specification. Sun Microsystems' collaboration with other leading vendors resulted in the Java Speech Markup Language (JSGF), possibly the first commercial attempt at such a standard.[40] More recently, the World Wide Web Consortium (W3C)[41] chartered the Voice Browser Working Group[42] to promote the Voice Web through the creation of standards, including a grammar format: *Speech Recognition Grammar Specification* (SRGS).[43]

---

[39] The SAPI interface to an ASR engine can be used by both application programmers and gateway architects, but in practice, ASR vendors provide their own programming language and offer SAPI compliance solely for the purpose of hardware integration. See http://www.microsoft.com/speech/technical/SAPIOverview.asp.

[40] Version 1.0 of JSGF was released on 26 October 1998 (http://java.sun.com/products/java-media/speech/forDevelopers/JSGF/).

[41] The mission of the World Wide Web Consortium, an industry standards body created in October 1994, is to develop common protocols that promote the evolution and interoperability of the World Wide Web. See http://www.w3.org/.

[42] See http://www.w3.org/Voice/ for an overview of the grammar format and the many other activities of the Voice Browser Working Group.

[43] On 25 January 2000, the Voice Browser Working Group decided to base SRGS on JSGF and requested that Sun formally submit its specification to the W3C. With only a slight change in name to *JSpeech Grammar Format*, the specification was tendered to the W3C on 5 June 2000 (http://www.w3.org/TR/jsgf/). The SRGS specification itself was released as a Working Draft on 3 January 2001(http://www.w3.org/TR/speech-grammar), with an estimated recommendation date of November 2001.

4.6. Distributed & Embedded Speech Recognition

Any device capable of telephony can rely on a remote gateway to provide ASR functionality; however, there are advantages to offloading all or part of the speech recognition task directly to the end user device.

- **Cost**. The telephony interface and ASR engine command a large part of the infrastructure budget, so the Voice Service Provider would prefer to offload some of this computation responsibility to the end user. In addition, the high recurring cost of telephony (in comparison to a packet data channel) conflicts with the need to keep the speech channel open and monitoring the caller from call start to finish – imagine the high connection throughput required and expense incurred if an ISP were responsible to monitoring every movement of the mouse and updating the pointer on the screen. This evolution of functionality to the edges of the network is consistent with the general trend of the Internet industry: keep the network interior simple and put the processing on the hosts.

- **Accuracy**. Even on a digital cellular network, significant signal degradation occurs to the low bit-rate speech coding and channel transmission errors. Additionally, the input signal is re-sampled repeatedly on it journey from consumer to gateway: from analog to digital at the handset, digital-to-digital at the multiple carriers handoffs, yet again at the telephony interface, and finally within the ASR Front-End – imagine trying to read a fax that has been sent to multiple machines in succession. These errors can be eliminated if all the processing required to produce the feature vector occurs within the handset itself: the resulting feature vector (a highly compressed form of the original speech signal) can either be furthered processed by an on-board ASR Back-End or forwarded across the carrier's error-protected data channel to a centralized back-end engine.

- **Integration Path.** The consensus in the communications industry is that all services (including telephony) will eventually be deployed over general-purpose data networks, and the next generation cellular networks will likely reflect this design philosophy, e.g. telephony and WML-browsing will both occur over the same wireless data channel.[44] As cellular carriers evolve towards this unified network design, the SIT architecture must take advantage of the increasing

computational capacity of handsets and the data-centric network to which they are attached. As previously separate programs (e.g. Voice and WML Web browsing) on the handset begin to share this common bearer service, they will potentially begin to drive the next generation of combined multimedia applications.

There are currently two types of ASR architectures that embody these design principles. *Embedded speech recognition* migrates the entire ASR engine to the handset. While embedded ASR has been successfully deployed in support of limited applications like voice-activated dialing, the computational and storage requirements for dynamic speech dialogs readily exceed the capacity limits of the current generation of cell phones. However, by moving only the ASR Front-End into the handset, the compact feature vector affords many of the same benefits listed above: lower bandwidth requirement, use of data channel not telephony channel, near-elimination of channel distortions, etc. This *distributed speech recognition* (DSR) approach extends the traditional ASR architecture earlier depicted in Figure 17 (page 50), both of which are reproduced below:

[44] Note that even though the carrier networks are packet-switched in this scenario, they are not necessarily designated as part of the Internet. Instead of a common-carriage policy, carriers may reserve these packet networks for its own customers in order to ensure the timely delivery of the packet telephony stream.

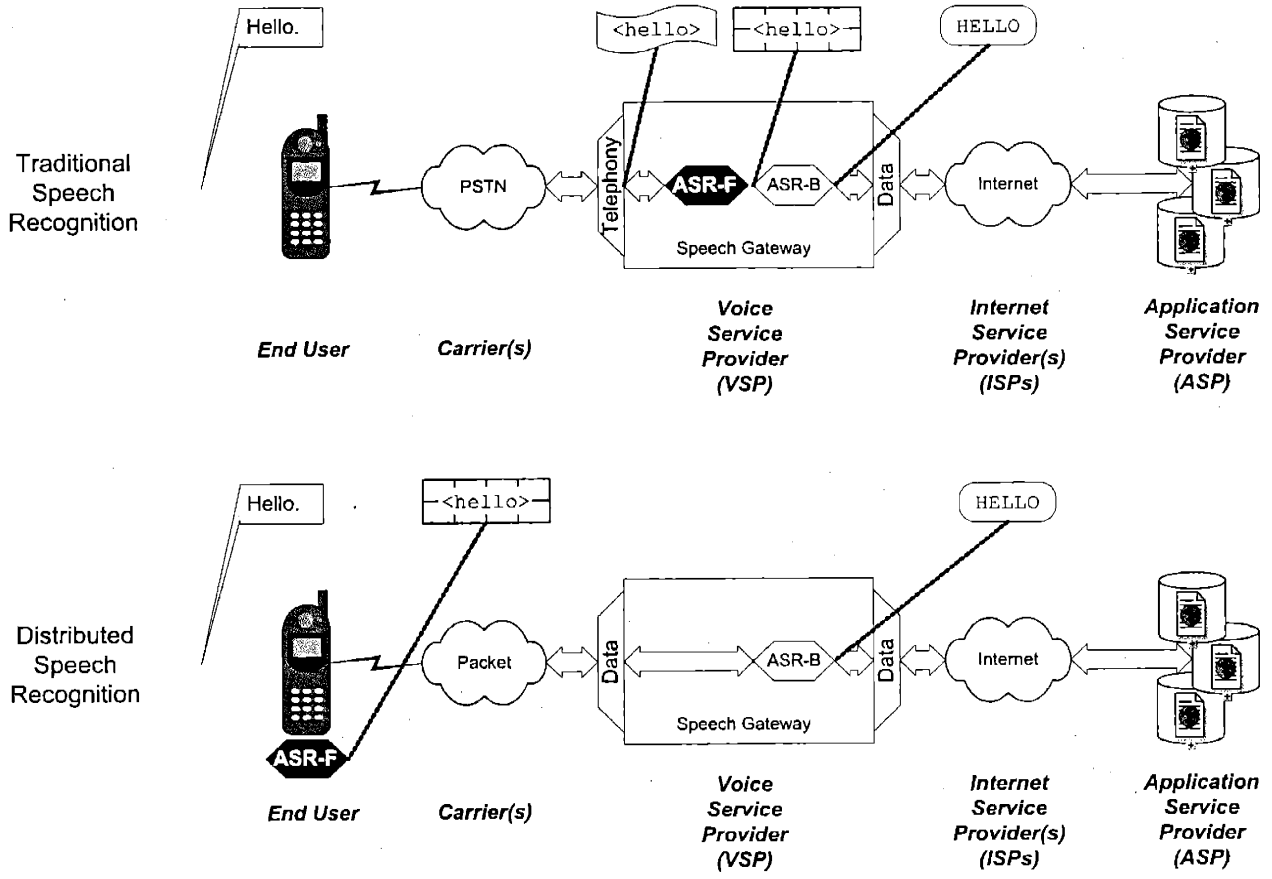

Figure 18. Traditional and Distributed Speech Recognition Architectures

The DSR approach transitions the digitization and pre-processing functions of the telephony card and the feature vector generation of the ASR Front-End to the end user device. This approach improves upon the packet-switched arhictecture illustrated in Figure 13 (page 45), since it not only takes advantage of the "packetization" of the carrier network, but also reduces the QoS burden by replacing the bandwidth-intensive packet telephony stream with the compact feature vectors.

From the perspective of this paper, the traditional ASR model is currently the only viable short-term architecture for supporting the VSP market. While boutique firms in the ASR industry have deployed the first iteration of DSR products, most commercial systems employ the traditional ASR architecture. In addition, the traditional model has the benefit of leveraging the existing base of regular wireless and wireline telephones, whereas DSR firms must negotiate the multi-year product cycles of handset manufacturers and carriers to incorporate their hardware and firmware technologies. However, in the medium-term, DSR (and to a lesser extent, embedded ASR) systems will play an important enabling role

for the Voice Web infrastructure, so it shall be included in the scenario analysis in Part II of this paper. Unless otherwise specified, the Reference Model for Speech-Interactive Telephony employs a traditional ASR and telephony card configuration.

Unlike a stand-alone ASR engine, the network distribution of the ASR Front- and Back-Ends requires considerable coordination in feature vector generation, compression, error-protection, etc. Since the carriers have no interest in being held hostage to a single ASR vendor's implementation embedded into both the consumer and network devices, the DSR industry has enlisted the non-profit European Telecommunications Standards Institute (ETSI)[45] to coordinate the development of key standards.

Within ETSI, the Aurora DSR Working Group[46] of the Speech Processing, Transmission and Quality Aspects Technical Committee (STQ)[47] has been actively developing and standardizing feature extraction and compression algorithms for the past 2 years. Published in April 2000, Aurora's first work item proposes an ASR Front-End that works well in low ambient noise environments.[48] The second and ongoing work item, slated for June 2001 release, will present other algorithms targeted at the more unforgiving mobile environment.[49]

4.7. Summary

While the foregoing discussion affords only a high-level view of this foundational technology, the reader should have an appreciation for the challenge of speech recognition:

- **Unconstrained input**. Unlike HTML forms that can rigidly restrict a user to select a single element within a finite set of options (e.g. drop-down list or radio button), callers can say anything they want at virtually any moment. While the caller can be trained or the application prompts be very explicit about the valid choices, they cannot ultimately restrain the speech input.

---

[45] See http://www.etsi.org.
[46] See http://webapp.etsi.org/tbhomepage/TBDetails.asp?TB_ID=438&TB_NAME=STQ+AURORA.
[47] See http://webapp.etsi.org/tbhomepage/TBDetails.asp?TB_ID=275&TB_NAME=STQ.
[48] RES/STQ-00018, *Speech Processing, Transmission and Quality aspects (STQ); Distributed speech recognition; Front-end feature extraction algorithm; Compression algorithms*, 11 April 2000. See http://webapp.etsi.org/WorkProgram/Report_WorkItem.asp?wki_id=9948.
[49] DES/STQ-00008, ETSI, (not yet published). See http://webapp.etsi.org/workProgram/Report_WorkItem.asp?wki_id=6402.

- **Grammars constrain the matching task.** Given the unavoidable fact of unconstrained input, the programmer can at least simply the task of recognition by providing a grammar specification for each dialog. Grammars condense the search space for a match, thereby improving responsiveness and accuracy.

- **Demands of a mass market mobile application**. In order to engage the widest possible audience, the speech gateway must support speaker-independent ASR technology that has been trained specifically for mobile use. It's additionally important to recognize that since markets both within and especially without the US are primed for voice services, a single speech gateway architecture that can support multiple languages is highly valued. Barge-in is necessary, especially for applications that feature long-play audio. Word spotting affords a measure of input flexibility by spotting keywords within larger fragments. Scoping of grammars is essential to implementing a multi-tiered application, particularly if it reaches across organizational boundaries. Runtime-compiled grammars enable fine-grained customization of an application for each caller in real-time.

- **Distributed Speech Recognition**. By moving the ASR Front-End into the consumers telephony device, the recognition accuracy and the potential for integration into other interaction modes (e.g. visual browsing) increase significantly. However, the DSR architecture is still in its formative stages of development and standardization, so virtually all speech gateway implementations employ a traditional monolithic ASR.

The previous chapter outlined how the speech gateway interfaces with the PSTN and Internet networks, and this chapter has illustrated how the incoming speech signal is rendered as computer-readable text. The next chapter explores the processes involved in responding to the caller: *output technology*.

Chapter 5. OUTPUT TECHNOLOGY-AUDIO PLAYBACK & TEXT-TO-SPEECH SYNTHESIS

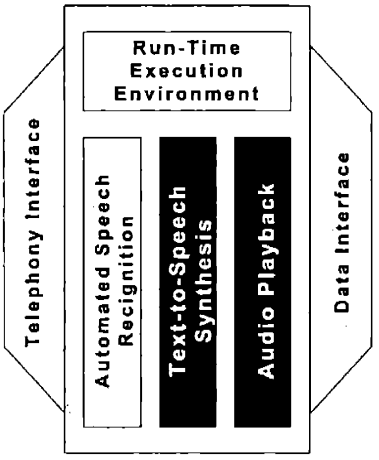

Figure 19. Speech Gateway (Currently on Output Technology)

As the name would suggest, there is an input "bias" evident in the discussion of speech gateway technologies. Great attention is paid to the role of automatic speech recognition, and for good reason: the unconstrained nature of the input is perhaps the most challenging and novel aspect of the system. But now the attention turns to the intended results of speech interaction: audio output.

5.1. Content Sources

Audio output is certainly no stranger to the SIT application community; however, the types of audio required by Voice Web applications bruise the current technology boundaries. The universe of audio for a traditional IVR application consists largely of short prompts that gently guide the user through the menus; more ambitious endeavors record longer files that convey canned information, e.g. computer-read bank balance. The common thread of these audio production efforts is that they were never meant for publication beyond the limited scope of the speech application, so the process of custom recordings is regarded as part and parcel of the programming effort.

In contrast, those firms that already produce audio content for traditional distribution channels view the Voice Web as a forward-looking opportunity to expand its publishing efforts at low marginal cost. For example, CNN's Headline News segment is originally produced for cable distribution, but it is converted

in real-time to a format more appropriate for HTML Web publication at very low cost. If CNN were presented with the opportunity to offer its content through a speech-interactive telephony application, it would certainly expect to leverage the existing HTML Web content to support its Voice Web publishing efforts. These audio segments differ from traditional prompts in many important respects - longer in duration, updated frequently, potentially live, published over many distribution modes (e.g. radio. HTML Web) – so the speech gateway needs to be capable of handling these various formats.

However, the wealth of branded, pre-recorded audio content already available cannot address all the information needs of the consumer:

- Live stock quotes: changes too often to be recorded in real-time
- Almanac: so expansive that the recording task would be astronomical
- Names in personal addressbook: specific to a particular consumer

Clearly, the speech gateway must not only negotiate the unfamiliar audio formats of the HTML Web, but also provide for the computer generation of "speech" from text sources.

5.2. Basics of Audio Output

As far as telephones and the PSTN are concerned, the output of the speech gateway is nothing more than the other side of the conversation. Human- or computer-generated audio alike is regarded simply as digital or analog signals traveling over the phone network. As explained in Chapter 3, the public telephone network's immutable internal format (8-kHz/8-bit/mono/μlaw PCM) requires that signals in dissimilar formats be brought into absolute compliance.

While the telephony interface is ultimately responsible for driving the signal onto the carrier network and plays a limited role in these compulsory conversions, the Text-to-Speech Synthesis and Audio Playback components abstract away the complexity and variety of audio formats, particularly those originally designed for Internet distribution.

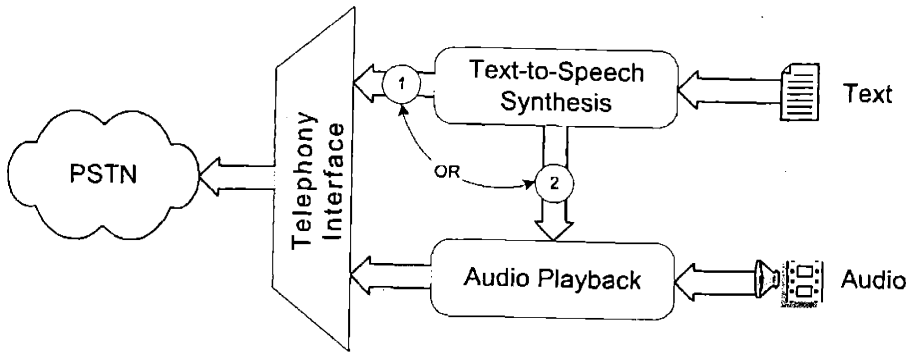

Figure 20. Output Technologies & Telephony Interface

5.2.1. Audio Playback

The most basic role of this module is to reconcile the audio format of the source with that of the PSTN (or at the very least, the internal format of the speech gateway).[50] For audio files created specifically for a particular speech gateway vendor, the format conversion task may be minimized or eliminated altogether; but for the literally hundreds of thousands of audio files already on the Internet in formats never intended for the PSTN (e.g. stereo, streaming), the real-time conversion activity is extensive. While this transcoding process cannot necessarily deliver the fidelity of the original audio source, it can abstract away this complexity from the programmer's perspective.

In addition, this component is also responsible for ensuring a consistent feature set across all audio formats. For example, suppose that the Playback Engine supports the conversion of two distinct file types: one that includes timestamps throughout the file to support features such as "rewind X seconds," whereas the other contains the audio data only. From the programmer and user perspective, the "rewind X seconds" command must be available for all supported audio types, so the Playback engine may optionally provide an internal buffer to implement a reasonable approximation of this rewind feature for this formerly "unnavigable" audio type.[51]

---

[50] The speech gateway's internal, "intermediate" format does not necessarily have to be match all of the PSTN format restrictions, but these formats are sufficiently similar to minimize the real-time processing required by the telephony card. Formats of this type include VOX, CELP, ADPCM, and WAV.

[51] For example, the popular ReplayTV (http://www.replaytv.com) and Tivo (http://www.tivo.com) systems bestow new features on standard television sets, such as limited rewinding and pausing of live broadcasts, by providing internal storage. Similarly, the Audio Playback extends the "feature set" of traditional audio file formats.

5.2.2. Text-to-Speech Synthesis

The Text-to-Speech (TTS) component is responsible for generating a speech signal from a text source, the mechanics of which are detailed later in the chapter. The resulting audio is represented on the system as an audio file not unlike the audio sources directly engaged by the Playback engine. Because of the fundamental similarity between the TTS output and the Playback module's traditional input, the TTS engine can be viewed either as a parallel or adjunct element, as Figure 20 illustrates.

Since the consumer would prefer not to remember two different sets of audio navigation commands for text and audio sources, functional consistency must be enforced throughput, lending support to the second schema. Additionally, centralizing all the integrative energies within the Playback module decouples the formerly tight integration between the telephony card and the TTS engine, affording more flexibility in choosing those vendors.

The next two sections detail the mechanics of these two output technologies, as well as their implications for the Voice Web and next-generation architectures.

5.3. Text-to-Speech Synthesis

5.3.1. TTS Architecture

If the task of ASR is to segment spoken phrases into phonemes for the purpose of text generation, TTS's role is simply the converse operation: to convert text strings into their constituent phonetic components for the purpose of speech generation. Indeed, many ASR and TTS engines share similar English-to-phonetic dictionaries and rules that drive pattern matching and speech synthesis. Contrast the following stylized view of TTS with the diagram of ASR in Figure 16 (page 50):

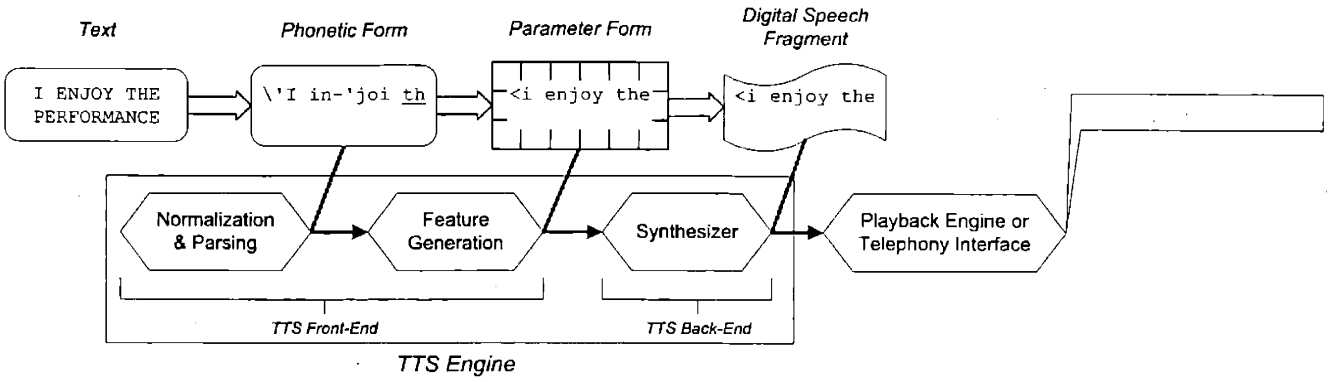

Figure 21. Basic Components and Example of Text-to-Speech Synthesis

1. **TTS Front-End**:

   a. **Normalization & Parsing**. The normalization phase identifies logical boundaries in the text fragment (e.g. sentences) and substitutes words for common abbreviations, acronyms, digits, etc. For example, "$1.25" becomes "one dollar and twenty-five cents." The parsing engine converts this normalized stream of text into the equivalent phonetic representation, along with the appropriate prosody characteristics, e.g. pitch amplitude, timing, stress patterns, etc.

   b. **Feature Generation**. This feature generation component converts the phonetic representation of the input text into a format appropriate for audio signal generation. These parameter values vary greatly across vendors, depending on the type of Synthesizer in the next step of the TTS process. For example, if the synthesis step relies on an acoustical model of the human voice, then the appropriate parameter values would include spectral and durational values, as well as more subtle elements such as head size, roughness, breathiness, etc. As in the case of the ASR engine, the compact parameterization of the text input presents the opportunity to distribute the TTS system across multiple locations and devices.

2. **TTS Back-End: Synthesizer**. The synthesizer, the component ultimately responsible for turning these text parameters into actual audio signals, is implemented using one of two competing technologies. *Waveform-concatenative* systems piece together speech from stored audio fragments of actual human speech; in contrast, *rule-based systems* depend on a general acoustical models of human speech in which the resulting audio signal is completely generated by the model itself and not pieced together from prerecorded fragments. Regardless of the synthesis methodology, the output of the synthesizer is a digital audio fragment.

3. **Playback Engine or Telephony Interface**. Whether the TTS engine is directly integrated onto the telephony interface or regarded as simply another audio file source (see Figure 20), the audio fragment produced by the synthesis component is eventually driven over the telephony network to the handset receiver.

Following the convention of the traditional ASR network infrastructure in Figure 17 (page 50), the TTS architecture is depicted below:

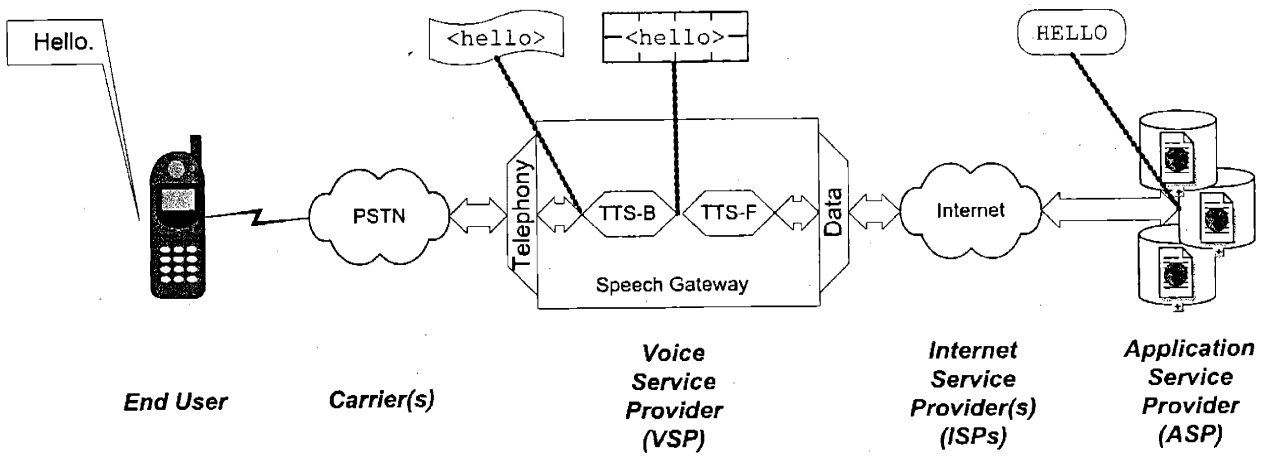

Figure 22. Traditional TTS Architecture

5.3.2. Speech Quality

For those consumers that have encountered TTS systems, the general impression is that while they produce *intelligible* results (i.e. the words being spoken are understood), they fail to reproduce the more subjective *pleasantness* of actual human speech. This failure to produce natural-sounding speech is an artifact of the complexity of spoken language generally. At first glance, it may appear that the TTS engine has a far easier job than the ASR engine: rather than working with inexact and unconstrained speech signals, the TTS system deals only with precise, unambiguous text strings - all the system has to do is look up the phonetic equivalent of each word and then just pull the sentence together. While it is true that the letters and words that constitute the elements of the text fragment are "known" perfectly, the pronunciation of the sentence relies on information that must be inferred using expert knowledge of the target language. The TTS Front-End is charged with this task of extracting these implicit parameters from the text fragment: rising pitch at the end of a sentence, commas that separate numbers (e.g. "1,234") do

not indicate a phrasal boundary so no pause is necessary, etc. Even if the Front-End executes its task perfectly, the TTS Back-End must accurately reproduce the speech using a necessarily imperfect acoustical model, one step removed from the true model of the human vocal tract.

Of the two methodologies for synthesizing speech signals, the concatenative approach currently produces more "pleasant" speech, at the cost of considerable computational and storage resources to engage and manipulate the repository of recorded speech fragments. In contrast, the pleasantness of a rule-based system is governed by the degree to which the model captures the perceptually relevant generalizations of actual human speech, the result of which is currently judged to be inferior in "pleasantness" to the concatenative approach. However, there are several compelling advantages of the rule-based schema, of particular interest to VSP infrastructure architects. First, these acoustical models are extremely compact and rely only on numerical parameter values to drive the synthesis process.[52] This engineering economy affords not only higher port densities on the VSP premises, but the potential for transitioning the TTS system in whole or in part to the telephony handset. In addition, given their generalized speech model, rule-based systems offer more flexibility in modifying the speech output, whereas a concatenative system is largely restricted to only the voices that were used to record the fragments. In an effort to balance the consumer desire for a pleasant-sounding voice with the VSP imperative for high port density, a few vendors are experimenting with hybrid concatenative/rule-based systems.

In practice, many voice application providers simply forgo the TTS process altogether in favor of a more "brute-force" approach for certain types of applications. In those cases where the text under examination cannot be anticipated (e.g. email), the reliance on TTS is unavoidable; however, if an application constructs its text output from a closed vocabulary, then recording each element in this finite "dictionary" becomes a possibility. In a sense, this is a form of application-level concatenated synthesis: instead of piecing together pre-recorded phonemes, the atomic elements can be words or even entire phrases. For example, a weather update is typically composed in a canned sequence whose elements are chosen from a finite set:

> "Today's weather will be" + "sunny/cloudy/hazy/drizzly" + "with a high of" + "(insert number)" + "degrees and a low of" + "(insert number)" + "degrees, and the barometer is" + "falling/rising/steady"

[52] As a point of reference, the L&H RealSpeak 2.0 TTS engine (http://www.lhs.com/realspeak/), which employs the concatenative approach, requires a dedicated server to provide only 24 concurrent channels of speech synthesis. In contrast, the ETI-Eloquence 5.0 TTS engine (http://www.eloq.com/products.htm) can run several hundred channels on a similar server configuration.

By carefully recording each of the audio elements above, the forecast can be simply pieced together by each fragment in sequence. This "hand-assembled" approach clearly creates additional complexity and effort for the application designer, but the frequent adoption of this approach suggests that TTS technology does not produce results acceptable to discerning consumers and providers. For example, TellMe (a consumer voice portal) and Zagat Survey (the publisher of the ubiquitous maroon restaurant guide) forged an exclusive alliance to create a speech-interactive telephony version of the rating service: in exchange for granting TellMe exclusive rights for telephone distribution of Zagat content, TellMe agreed to produce an a fully-recorded application.[53] Considering that there are over 15,000 reviews in the United States alone, and the average review is about 60 words (including address and ratings), TellMe has invested a sizable effort to avoid using the TTS technology that it provides on its own platform.[54] This vote of "no-confidence" for TTS technology in a production service may be indicative of the reticence that application providers feel towards computer-read text.

5.3.3. Standardization

Similar to the ASR engine, TTS technology is regarded from the perspective of both platform integration and programmer control. Several TTS vendors extol the virtues of SAPI compliance, but as before, this only marginally streamlines the task of the speech gateway architect. Application developers traditionally access the TTS engine through simple function calls from the ASR vendor-supplied programming language. The traditionally integrative function of the ASR language and the evolution of the execution environment will be discussed further in a later chapter.

However, there is yet another language that can be used to engage the TTS engine directly: speech synthesis markup language. Just as the programmer supplies dialog-specific grammars to guide the ASR process, a speech synthesis markup language enables the insertion of annotations to guide the textual analysis and even the actual speech generation process. Since each synthesis language exerts fine-grained control over the TTS engine, both the syntax and functionality vary greatly across vendors. The very same Sun Microsystems' collaboration[55] that produced the Java Speech Grammar Format also resulted in the

---

[53] See http://www.tellme.com/menu/entertainment/entertainment_restaurants.html.

[54] From the TellMe Technical Overview: "Pre-recorded audio is almost always preferred in production applications, as it provides the most natural sounding interface. Text-to-speech is useful for rapid application prototyping, and for generating output from data whose content cannot be guessed in advance. For example, an application that reads a user's e-mail over the phone." See: http://studio.tellme.com/faq/techoverview_voice_faq.html.

[55] The creators of Sun's Java Speech Markup Language borrowed heavily from the following previous proposals:

first commercial, unified annotation standard: *Java Speech Markup Language* (JSML).[56] And once again, the W3C's Voice Browser Working Group has assumed the leadership role by creating its own TTS annotation standard: *Speech Synthesis Markup Language* (SSML).[57]

SSML is designed to guide the structural and production elements of any TTS engine:

TEXTUAL (A.K.A. STRUCTURAL) ELEMENTS

- **Acronym**: "U S A"

  ```
  <say-as type="acronym"> USA </say-as>
  ```

- **Numbers:** "One two three four five"

  ```
  <say-as type="number:digits"> 12345 </say-as>
  ```

- **Dates:** "March thirtieth nineteen twenty two"

  ```
  <say-as type="date: mdy"> 3/30/1922</say-as>
  ```

- **Currency**: "Forty dollars and twenty-two cents"

  ```
  <say-as type="currency"> $40.22 </say-as>
  ```

- **Email:** "Bob at hotmail dot com"

  ```
  <say-as type="net:email"> bob@hotmail.com </say-as>
  ```

SYNTHESIS (A.K.A. PRODUCTION) ELEMENTS

- **Gender**: If available on the platform, use the voice of a man

  ```
  <voice gender ="male"> I am a man </voice>
  ```

- **Emphasis:** "I can't believe you **ATE** that."

  ```
  I can't believe you <emphasis> ate </emphasis> that
  ```

---

1. Edinburgh University. Taylor, P. A. and Isard, A., *SSML: A speech synthesis markup language*, Speech Communication 21 (1997) p. 123-133.
2. Massachusetts Institute of Technology. Slott, J, *A General Platform and Markup Language for Text to Speech Synthesis*, MIT Masters Thesis, 1996.
3. SABLE Online Community: Sproat, R. et. al., *SABLE: A Standard for TTS Markup*, 5th International Conference on Spoken Language Processing, Sydney, Australia, November, 1998.

[56] JSML version 0.6 (Beta) is available at http://java.sun.com/products/java-media/speech/forDevelopers/JSML/index.html. In

[57] On 14 December 1999, the Voice Browser Working Group decided to base its SSML specification on JSML and requested that Sun formally submit its specification to the W3C. With only a slight change in name to *JSpeech Markup Language*, the specification was tendered to the W3C on 5 June 2000 (http://www.w3.org/TR/jsml/). Perfectly tracking the release schedule for the W3C's Speech Recognition Grammar Specification, SSML was released as a Working Draft on 3 January 2001(http://www.w3.org/TR/speech-synthesis), with an estimated recommendation date of November 2001.

- **Prosody:** Change the pitch, timing, and/or volume for effect

```
<prosody pitch="low" rate="slow"> Luke, I am your father
</prosody>
```

5.3.4. Distributed & Embedded TTS

The motivations for evolving the components of the ASR engine – cost, accuracy, and consistency with migration to data-based networks – apply equally well for the TTS engine. As before, the general rule is to minimize the bandwidth required between the handset and the speech gateway, while recognizing the computational and storage constraints of the handset itself. A TTS engine fully embedded within the handset would be ideal in that the compact, original text could be sent over the data network directly; however, the robust generation of the phonetic and parametric forms for arbitrary text strings is a challenging task for an embedded processor, particularly if the "pleasantness" of the resulting speech is an important consideration. An alternative to the embedded approach is to transition only the signal generation element (TTS Back-End) to the handset, taking advantage of the economy of the parametric form. Since the concatenative methodology requires extensive storage for pre-recorded audio fragments, only the rule-based approach is a candidate for the distributed architecture depicted in Figure 23 below:

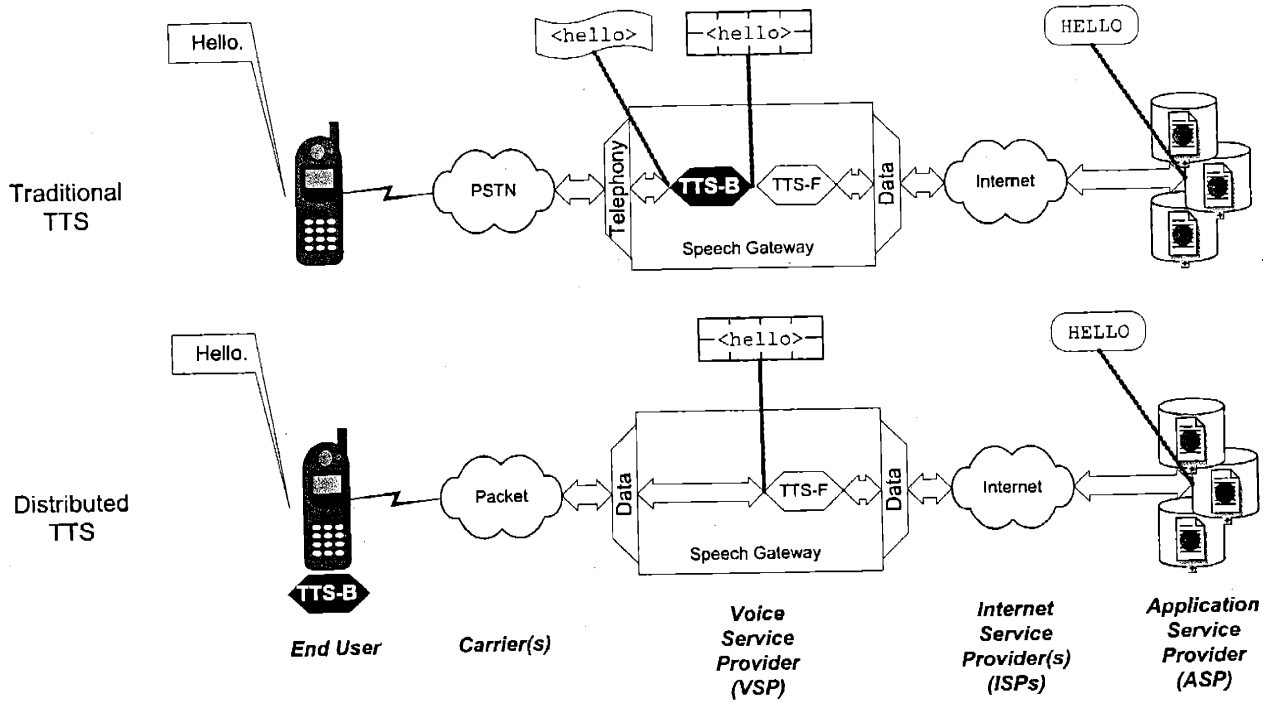

Figure 23. Traditional and Distributed TTS Architectures

Regardless of the architecture, the quality of the generated speech is ultimately the only relevant characteristic for the consumer, so the pleasantness of rule-based systems must improve significantly before application service providers adopt text-to-speech generation as a standard part of the implementation process. In addition, there is no TTS counterpart to the Aurora DSR Working Group within ETSI or any other organization, so there are no standards that currently govern the interface between the TTS Front-End and Back-End – all of the industry's energies are focused on fully-integrated systems (whether traditional or embedded). As in the case of ASR, this paper regards the traditional TTS architecture as the only viable model in the short-term, so the distributed and embedded models will be relegated to the Scenario Analysis towards the end of the paper.

Since TTS element ultimately produces audio waveforms, its output can be regarded as simply one among the many supported by the audio playback module, to which the rest of this Output Technology chapter is devoted.

5.4. Audio Playback

Unlike a traditional IVR application in which only certain pre-recorded audio formats are supported and the audio files must be stored proximal to the application code, the Web model introduces several innovations to the audio playback process. Just as a single HTML document can refer to elements stored anywhere on the Internet, a Voice Web program must also be capable of accessing audio files from any remote location. This mandate for universal access, coupled with the desire to repurpose existing audio content on the Internet, necessitates the flexibility to process the heterogeneous audio types commonly encountered on the Internet. Since most of the Internet's audio repository designed for streaming, the speech gateway must be able to engage and reformat these streams for PSTN playback in real-time. Clearly, expanding the audio playback element's functionality for the Voice Web model is not a trivial task.

5.4.1. Audio Types

Audio intended for the Web market, and the systems that convey them, assume a packet-switched network, whereas the current delivery mechanism for Voice Web content is over the circuit-switched telephony network. This mismatch of engineering considerations potentially leads to significant

degradation in audio quality and performance. In order to illustrate these differences, a vocabulary is required to distinguish the various audio types encountered:

- **Live vs. Pre-recorded Audio**. Playback for *pre-recorded* audio can occur only after recording is completed; whereas *live* content can be engaged during the recorded process, enabling playback only moments after the signal is created. The difference between live and pre-recorded content may make no difference to the listener, but weigh heavily on the engineering considerations of the system.

- **Streaming vs. Static Audio**. These terms are often used interchangeably with *live* and *pre-recorded*, respectively, which tends the obscure the distinction. *Live* and *pre-recorded* describe the temporal overlap between the start of playback and end of content creation, whereas *streaming* and *static* relate to the temporal overlap between the start of playback and end of transmission. A static audio file must be transmitted completely before playback can begin, but a streaming audio source can begin playback before the end of transmission. The audio source sends a sequence or "stream" of audio data that can be acted upon incrementally by the audio recipient: the original audio signal is reconstructed in real-time as data arrives at the destination.

By definition, *live* content must be streamed since playback must begin before the end of content creation and the end of transmission (e.g. sometimes there isn't even a end of transmission, such as a continusouly operating radio station), but pre-recorded audio can either be streamed or downloaded in its entirety before beginning playback.

Whereas traditional speech applications utilize pre-recorded and static audio prompts, the audio files commonly encountered on the Web were designed for a very different environment. Since the Internet offers no guarantees on throughput and most consumers have very low speed access (e.g. less than 56-kbps), a static audio download could potentially take a very long time and playback could not begin until the entire file had been transmitted. At the same time, since the Internet offers no provision for QoS packet delivery, a streamed file would afford the advantage of lower playback latency but must contend with the vagaries of delayed, dropped, or corrupted packet delivery on the Internet. In response, audio vendors created audio file formats and communications protocols that include error recovery, redundancy, prediction, and compression schemes that account for the "lossy" and "bursty" packet delivery service of the Internet. Depending on the application and Internet link, these streaming technologies sacrifice a level of audio quality in order to maintain the audio feed under less-than-ideal transmission conditions.

Technically speaking, audio data is "streamed" all the time within a computer system, such as between the hard drive and the audio card, but the term is commonly understood to describe the concurrent, incremental movement and reassembly of an audio signal over a variable bit rate and lossy packet network, such as the Internet. While an audio file can certainly be streamed across a private network, such as a LAN connecting a server and a speech gateway, the private network can be engineered to avoid the characteristically "hostile" operating environment of the Internet.

5.4.2. Elements of Audio Streaming

Since streaming audio is the only novelty necessitated by the Voice Web model, the attention turns to the mechanics of these audio playback systems. Independent of the vendor, an audio streaming system consists of the following three elements: *producer*, *server*, and *player*. The producer is a provider-side product that provides batch or real-time encoding, compression, and segmentation of audio feeds, with the net result suitable for Internet streaming. With the exception of producer for live streams, the server and player elements are the active components during audio playback.

In a traditional HTML audio architecture, the server module resides proximal to the content store and the player module is located on the consumer's computer. Typically, this player element is a freely-downloadable software package that integrates with the user's HTML browser. This logical flows persists in the SIT architecture but the mechanics are modified slightly: the server element remains adjacent to the audio source but the player module resides within the VSP's speech gateway, not within the consumer's handset. The location of the player technology is entirely consistent with the unique service provider-side execution model of the SIT architecture.

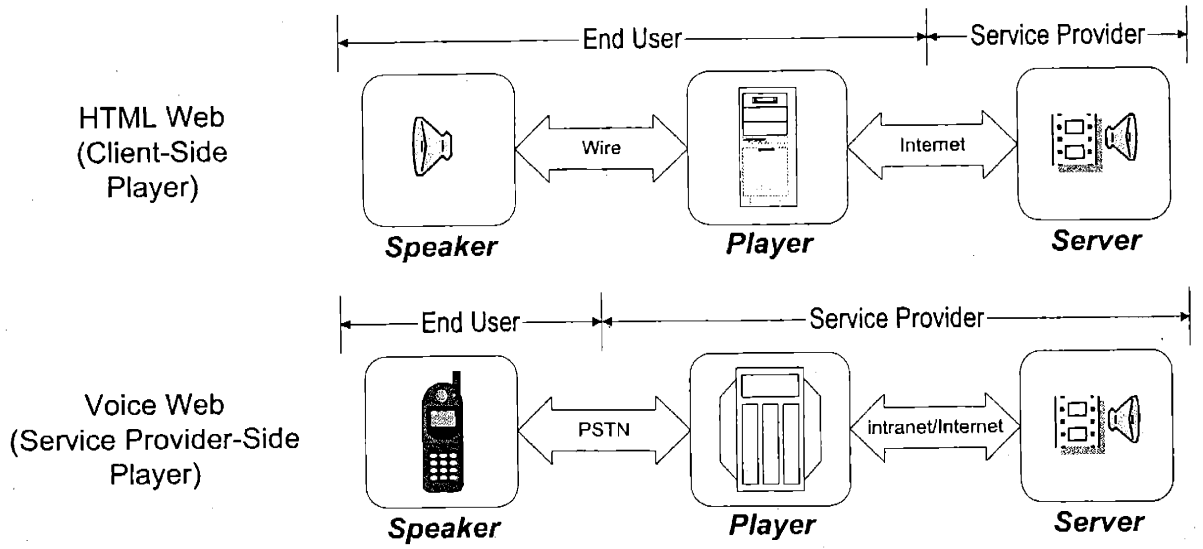

Figure 24. Client-Side vs. Service Provider-Side Streaming Audio Architecture

As usual, the telephone can be regarded simply as a speaker and microphone pair with a long "cord" to the speech gateway, where all the processing for audio resides. In the vocabulary of the SIT architecture, the streaming audio player forms the core of the Audio Playback module in the VSP's speech gateway, and the streaming audio server resides on the ASP's general serving infrastructure. The player module basically performs the inverse operation of the server (e.g. decompression, reassembly, interpolation, etc.), so this puts another computational load on an already stressed speech gateway. The obvious challenge is to integrate the streaming functionality into the speech gateway while maintaining density and scalability targets.

5.4.3. Standardization

The decentralization of the single audio playback function into two interworking components across the Internet closely resembles the architecture of distributed ASR and TTS. In this case, the level and type of standardization in the audio streaming arena provides an example of how standards organizations can fail to drive the evolution of a Web technology. In October 1996, RealNetworks and Netscape Communications jointly submitted the *Real Time Streaming Protocol* (RTSP) to the Internet Engineering task Force (IETF),[58] an open and influential international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture. After extensive co-

[58] See http://www.ietf.org/. The IETF is an "organized activity" of the Internet Society (ISOC): http://www.isoc.org/.

development of the draft by the creators, Columbia University and the IETF MMUSIC Working Group,[59] RTSP was published as a Proposed Standard in April 1998.[60] A proposed standard is generally considered as the stable result of thorough review and interest, but lacks the imprimatur of the Internet standards community.

In advance of any formal blessing of RTSP, RealNetworks developed a real-time streaming platform for HTML Web service providers and consumers: RealSystem. Building upon the general RTSP protocol, RealSystem is a suite of software modules that implements the streaming architecture defined earlier: RealSystem Producer, RealSystem Server, and RealPlayer.[61] As its products have evolved to take advantage of new technologies and service offerings, RealNetworks has introduced proprietary transport protocols (e.g. Real Data Transport[62]) and codecs (e.g. RealAudio[63]), with the result that only RealSystem components are capable of creating, serving, and playing RealAudio files. In addition to supporting RealAudio streams, the RealPlayer offers the convenience of engaging traditional static audio formats such as AU and WAV.

Another major platform for audio streaming is Microsoft's *Windows Media Technologies* (WMT), the components of which closely parallel the reference model: Windows Media Encoder (i.e. producer), Windows Media Services (i.e. server), and Windows Media Player.[64] As with RealSystem, WMT utilizes proprietary protocol and codecs (Windows Media Audio or "WMA"), creating a similar monopoly on creating, serving, and playing the audio streams. The Media Player is available on a variety of end user devices, including desktop and handheld computers, and flexibly supports several popular static audio formats.

There are certainly other audio platform vendors (e.g. Apple's QuickTime), but RealSystem and Windows Media are the *de facto* streaming standards of the WWW audio landscape, with one notable exception: MPEG-1/MPEG-2 Layer III Audio or, more commonly, *MP3*. The Motion Pictures Experts

---

[59] See http://www.ietf.org/html.charters/mmusic-charter.html.
[60] RTSP is published under Request for Comments 2326: http://www.ietf.org/rfc/rfc2326.txt.
[61] For an overview of RealNetworks' RealSystem, see http://www.realnetworks.com/devzone/documentation/wp_g2_overview.html.
[62] RealNetworks' proprietary Real Data Transport (RDT) protocol is used for the data connection and retransmisstion of lost packets, whereas RTSP is given the task of controlling communications. The specification is obviously closely held, but brief mention of RDT can be found in the whitepaper *RTSP Interoperability with RealSystem Server 8*, 7 December 2000 (http://docs.real.com/docs/rtsp.pdf).
[63] The RealAudio codec (compression/decompression specification) is another closely-held element of RealNetworks' proprietary streaming audio technology. A general description is available at http://www.realnetworks.com/devzone/documentation/wp_g2_music.html.
[64] For a general introduction to the WMT technology suite, see http://www.microsoft.com/windows/windowsmedia/.

Group (MPEG),[65] a working group of the International Organization for Standardization (ISO)[66], develops codecs for moving pictures and audio, of which MP3 is an example. MP3's compression algorithm significantly reduces the size of stereo audio without seriously impacting the signal fidelity, so it has become the format of choice for storing and sharing CD-quality music on the Internet. MP3 files can be downloaded like traditional static files, or streamed provided that the remote server supports the Real-time Transport Protocol.[67] Unlike RealAudio and WMA files, the codec for MP3 is public, so there are a variety of platforms capable of creating, serving, and playing these files, including RealSystem and Windows Media.

5.4.4. Integration

Outside of CD-quality audio recordings, the RealAudio and WMA formats completely dominate the WWW audio space, so the task of the speech gateway designer is to integrate either or both of these technologies, and to ensure that the gateway's architecture is compatible with the streaming methodology internally, e.g. the TTS engine can stream audio directly to the Audio Playback element, particularly useful for long text sequences. The speech gateway architect should be legitimately concerned with the impact on scalability and density by integrating these player technologies, originally intended for the end user desktop and laptop computers.

However, there are several advantages to building the Playback module around these third-party streaming engines. First, the player provides a layer of abstraction for a variety of audio formats: instead of having to engineer support for several audio types individually, a single instantiation of a RealAudio or WMA engine can support not only its own proprietary streaming format but also a wide array of popular static formats (including WAV and MP3). Second, it provides a consistent navigation interface to all audio formats, so the telephony voice user interface can simply tie directly into the built-in feature set, such as "stop," "rewind," "pause," etc. Third, the reliance on an established streaming vendor affords a measure of protection against technological obsolescence, since it is the responsibility of RealNetworks and Microsoft to keep abreast of advances in audio compression, add support for other audio formats, and

[65] See http://www.cselt.it/mpeg/.
[66] See http://www.iso.ch/.
[67] The Real-time Transport Protocol (RTP version 2) provides end-to-end delivery services to support applications transmitting real-time data (e.g., interactive audio and video) over unicast and multicast network services. RTP is defined in IETF RFC 1889 (http://www.ietf.org/rfc/rfc1889.txt), and has remained at the level of Proposed Standard despite its wide application, including acting as the delivery mechanism for RTSP.

evolve the product to support new functionalities.[68] Finally, the player provides a natural point at which to apply the post-processing necessary to modulate the audio signal in a manner consistent with the PSTN's engineering assumptions. While the playback element certainly cannot eliminate the signal degradation caused by this format conversion, the player can enforce a consistent normalization and encoding process across all file types, including the output of the TTS engine.

While streaming audio support is certainly atypical of a traditional IVR system, the configuration of the traditional stream audio architecture for the HTML Web is well understood. While the Audio Playback element performs tasks beyond packet decompression and reassembly (e.g. normalization, dialog logic interface), the integrated audio engine forms the core of the Playback module. Combining the elements of both the speech-interactive telephony and streaming audio architectures yields the following:

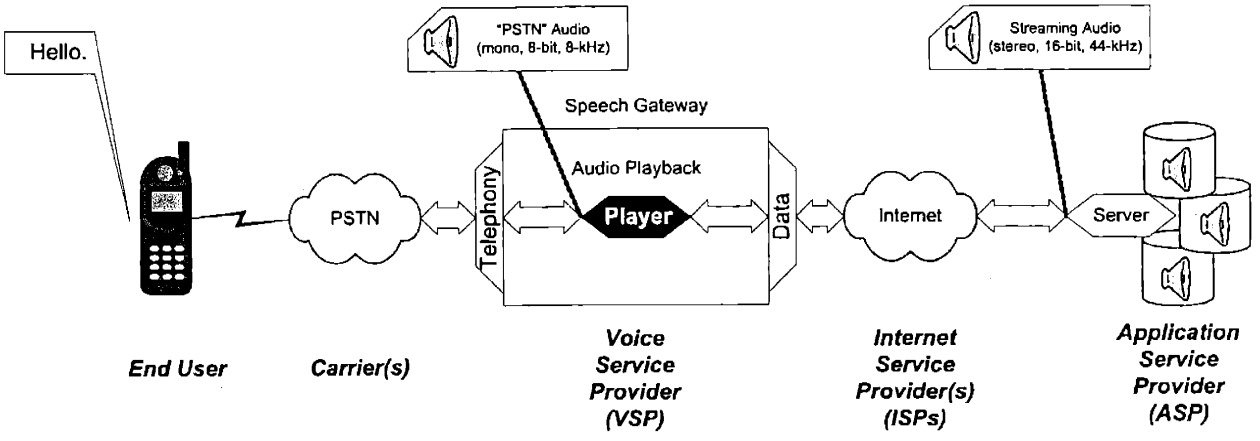

Figure 25. Audio Playback Architecture

5.4.5. Next Generation Architecture

Unlike the distributed ASR and TTS architectures that reside on the handset and speech gateway, the streaming audio architecture is shared between the VSP's speech gateway and the ASP's servers; however, player components have already been developed to reside directly on the handset. The RealSystem[69] and Windows Media Format[70] Standard Developer Kits (SDKs) are available as thin clients

[68] For example, RealNetworks offers an Advertising Extension product that allows the insertion of audio ads into the primary audio stream (http://www.realnetworks.com/products/servers/advertising). By integrating the RealSystem into the speech gateway, the service provider can tap into an existing revenue stream without having to develop the technology itself.

[69] See http://proforma.real.com/rn/misc/g2sdk/.

for direct integration onto portable music devices and digital assistants, e.g. the RCA Lyra2 player supports all the major streaming formats, including MP3, WMA, and RealAudio.[71] In addition to software clients, player engines are being ported directly to special-purpose DSPs to ease the integration task for handset and portable device manufacturers, paralleling similar trends in the embedded ASR and TTS industries, e.g. Texas Instruments produces embedded DSP versions of the WMA and RealAudio player engines, targeted at next-generation portable device manufacturers.[72] Regardless of whether the software or firmware implementation is selected, the transition of the player component to the handset is consistent with the distributed architectures presented earlier:

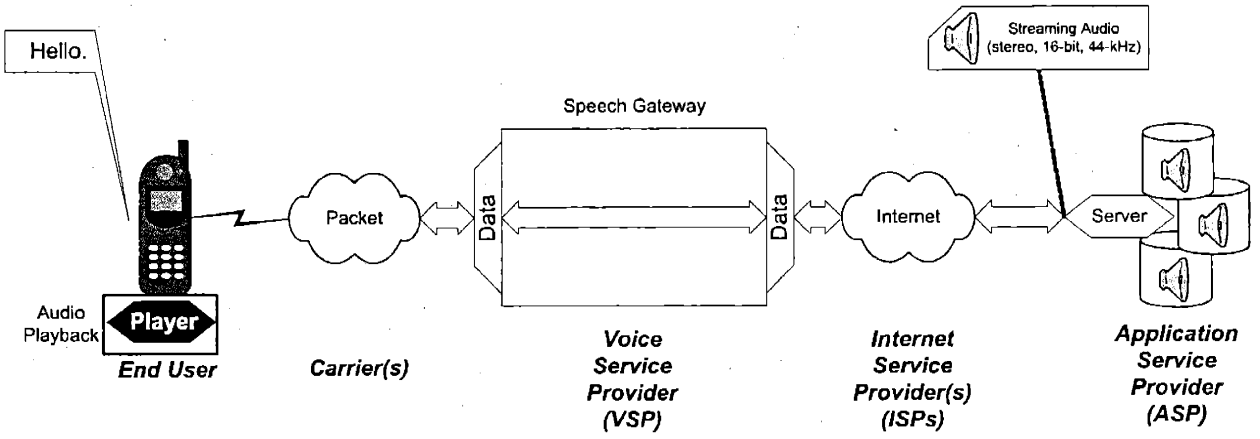

Figure 26. Next-Generation Audio Playback Architecture

Apart from offloading from the speech gateway the computational load of decompressing and converting audio files in real-time, the end-to-end streaming of content files creates the possibility for a high-fidelity audio experience impossible over the PSTN. Provided that there is sufficient bandwidth between the end user's handset and the ASP's servers, the level of media complexity is unbounded. For example, CD-quality MP3 files can be streamed in real-time to a handset outfitted with a stereo headphone jack – even synchronized audio and video is possible with the inclusion of a screen. The requirement for QoS end-to-end over multiple networks is not a trivial consideration and will be one of the topics addressed in the subsequent chapter on Enhanced Network Services, but the handset and player elements are not the limiting technologies in this case.

[70] See http://msdn.microsoft.com/workshop/imedia/windowsmedia/sdk/wmsdk_format.asp.
[71] See http://www.rca.com/product/viewdetail/0,1322,PI45033-CI100009,00.html.
[72] See http://www.ti.com/sc/docs/news/2001/01016.htm and http://www.ti.com/sc/docs/news/2001/01045.htm.

A cursory examination of Figure 26 should reasonably encourage the reader to question the role, if any, of the Voice Service Provider in the next-generation Voice Web architecture. Looking back to Figures 18 (page 68) and 23 (page 80), the distributed ASR and TTS architectures still require the computational and storage capacity of the VSP network to drive the SIT application. In addition, the integrative role of the execution environment itself has not yet been discussed, so the telephony voice user interface implemented on the speech gateway must be able to control the audio playback component, regardless of its location in the data flow. The long-term vision of the system, which implies a complete transition of the speech gateway's functionality to embedded subsystems in a handset, may indeed suggest the disappearance of the VSP's role in the Voice Web process, but this paper is more concerned with the short- and medium-term structure of the SIT architecture.

5.5. Summary

This chapter examined the sources of content for Voice Web firms and the technological implications for speech gateway vendors. This focus on the content and formats currently deployed on the Web is not necessarily indicative of the audio requirements of all speech applications. For example, consider a traditional, standalone IVR application that requires custom recording of all prompts and relies on the legacy telephone network for distribution; in this case, there is clearly no interest or need to support streaming audio formats. Recalling the differentiation between forward-looking and inward-looking voice application firms, currently only the former are interested in driving the support for universal audio access. The inclusion of a real-time audio streaming engine potentially adds considerable complexity and cost to the speech gateway (depending on whether the gateway's architecture is amenable to internal audio streaming), so the forward-looking Voice Web market must provide compelling economic incentives to the gateway vendors in order to add this functionality.

After having reviewed the individual components of speech-interactive telephony – interface, input, and output technologies – the attention naturally turns to the element that integrates these disparate functionalities into a single network service: the Execution Environment.

Chapter 6. EXECUTION ENVIRONMENT - VOICEXML INTERPRETER

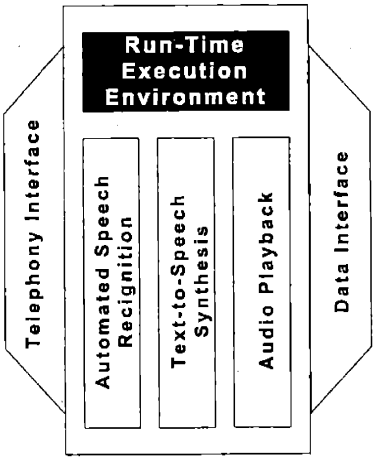

Figure 27. Speech Gateway (Currently on Execution Environment)

In a speech gateway platform, the run-time execution environment can be politely considered as the body of a many-headed hydra: its role is to assimilate the various interfaces and subsystems into an integrated, functional whole. If the speech gateway is properly implemented, this will be the single direct point of interaction between the programmer and the entire system. Like any other operating system (OS), it is responsible for system monitoring, error reporting, message passing, error-checking, multitasking, I/O interrupt handling, memory allocation, and perhaps most importantly, program execution and state management. The execution environment can be viewed as a virtual machine running across many physical elements, each of which may have its own operating system, e.g. Windows NT. Since the guidelines for designing such an environment do not differ notably from general OS design principles for scalable systems, matters of implementation are outside the scope of this paper.

The execution environment is the middleware between the individual components and the programmer, and this chapter explores the structure of the interaction between the system and the application developer, the nature of which will have far-reaching consequences for the viability of the Voice Web. More specifically, the language that is used to convey the business logic is a natural anchor point around which to build enthusiasm and momentum for the Voice Web concept. The next section explores the unique language attributes required by Voice Web applications, followed by an examination of whether recent trends in the industry move towards or away from these guiding principles.

6.1. Programming Language Attributes Required for Voice Web Applications

While the technologies discussed thus far do indeed present formidable engineering challenges, they represent enhancements of a matter of degree and not kind. For example, speaker-independent ASR has been around in IVR systems for quite some time and has been successfully deployed in mission-critical enterprise applications, streaming audio involves integrating products that are already available to the HTML Web community, etc. However, a language capable of supporting the Voice Web model marks a significant departure from traditional IVR languages. As will be demonstrated, many of these attributes are borrowed liberally from the experience of dynamic applications on the HTML Web.

6.1.1. Interpreted (Not Compiled) Language

The principal purpose of any computer language is to facilitate the expression of the programmer's directives into machine-readable code (i.e. machine language). Languages may vary according the level of abstraction from the underlying computer architecture; regardless, one of the following two events must occur prior to execution:

- The code must be translated or *compiled* into machine language, which can be directly executed on the target platform.
- The program will be *interpreted* by yet another program already running on the target platform. This secondary program, appropriately called an *interpreter*,[73] executes the program on behalf of the platform.

The classic example of an interpreted system is the pairing of the web browser and the HTML document: A team of programmers at Netscape (a subsidiary of America Online) is responsible for the Navigator HTML browser product. The program itself is a classic example of a compiled program: the team composed the code that details the operation, and then compiled the program into code that will run only on specific platforms: Windows, Linux, Macintosh, etc. One cannot purchase Navigator for Macintosh and install it on a Windows computer: the machine languages are completely incompatible. However, the web documents themselves are platform-independent, e.g. an HTML document, replete with graphics and text, should appear and function the same regardless of whether it is interpreted by Navigator for

[73] Most interpreters are implemented as compiled programs.

Macintosh or Navigator for Windows. The HTML documents are not translated into machine code, nor are they executed directly by platform - they are executed or *interpreted* by the Navigator program itself.

Compilation affords the benefit of efficiency while sacrificing malleability. For the scale and complexity of most computer applications, a program will generally run more quickly in compiled form than in an equivalent interpretable form. The compilation process occurs offline, leaving only the task of machine-level execution at run-time. However, compilation does not lend itself well to situations where the program code itself is changing and the interaction with the user is highly modular. The compiled program methodology works well for those speech applications that traditionally eschew dynamic call flows and user customization, but the Voice Web's emphasis on dynamic media content and consumer personalization requires an interpreted language's support for real-time dialog generation and back-end integration with data repositories. In addition, since the Voice Web is an open network over which any speech application can be published, it would be impractical to require the compilation of each program to every possible platform.

6.1.2. Portability across Speech Gateways[74]

Code portability across speech gateway platforms is of interest to both traditional IVR application providers and Voice Web firms. Even an ASP with the wherewithal to purchase and operate its own VSP network prefers the flexibility of changing its gateway vendor without modifying or recreating the application code. Considering the number of possible combinations of ASR, TTS, and Playback engines within a single instance of a gateway, the speech language provides a consistent functional interface across configurations and vendors. In addition to reducing some of the risk of being tied to a particular vendor, there is a real-time publishing component unique to the Voice Web: since the code will potentially be interpreted on a variety of speech gateways, the programmer cannot be burdened with generating code appropriate for each platform. An agency focused on media publishing desires the widest distribution without also requiring substantial reengineering for each channel - a multi-platform dialog language is essential to creating this unified distribution network over a competitive gateway industry.

[74] To clarify, the author does not consider a single ASR firm porting its own engine to multiple gateway platforms as creating a truly multi-platform language, since the programmer is still beholden to the particular interests of a single technology vendor. In other words, a Windows application may operate on both a Dell and Gateway computer system, but an HTML document can execute without modification on a wide array of compliant hardware and software platforms, demonstrating the true meaning of the term "portable."

6.1.3. Open Standard

The success of products like the Linux operating system and the Java programming language has created a cultural presumption that an effort created and supported by the entire community is inherently good, but there are many "closed" standards that enjoy considerable commercial success: RealAudio, Solaris OS, Palm OS, ATM, C/C++, etc. Many of these standards come from companies that, in addition to reselling the component technology, offer a vertically integrated product, e.g. 3Com licenses the PalmOS to outside PDA vendors but also sells its own line of PDA devices. While it is impossible to pinpoint exactly why these proprietary standards have enjoyed commercial success where others have failed, there is one obvious aspect they share: the commitment to multi-platform portability.

An open standards process cannot guarantee that a language will be supported on multiple platforms, but it ensures that its development will not be beholden to the interests of a selected few firms. In moving from a vertically integrated to a tiered market structure, the input of agencies at every level is necessary for building a vibrant, competitive application and platform industry. In an industry riddled with incompatibility, an open-standard community may be the necessary antecedent to developing an effective multi-platform speech language, one which has the confidence of ASR vendors, speech gateway developers, programmers, etc.

6.1.4. eXtensible Markup Language (XML)

Thus far, the case has been made for a speech programming language that is interpretable (not compiled), natively supported on multiple (vendor) platforms, and secures the technological and financial confidence of the speech community through an open-standard effort. Since these values represent a significant departure from the character of most speech languages, it would be a significant advantage to plug into a pre-existing developer community that propagates these values for other industries. Namely, the eXtensible Markup Language (XML) provides a fertile foundation upon which to build a new language that achieves these principles.[75]

XML is formally defined as a *meta-language*: a "self-describing" data interchange format that describes content, not format or layout. In conjunction with a style- or layout-sheet, a well-defined meta-language can be used to create a variety of new markup languages. To give a loose example using human

[75] Please refer to http://xml.org/ for additional information about XML.

languages, a meta-language would define generic "containers" such as nouns and verbs, and the derived language could define the universe of nouns and verbs, as well as how they should be strung together or "displayed." Perhaps the most familiar example of an XML-derived language is HTML, the standard document format upon which the World Wide Web is built; however, there are literally dozens of XML languages that have been defined for a variety of Web applications and industries: advertising (adXML), wireless web documents (WML), gene expression (GEML), music (MusicML), and even food recipes (DESSERT).[76]

The powerful idea behind XML is that a single interpreter should be able to support multiple markup languages. Rather than each application having to create an interpreter for a particular document format (e.g. gene expression and advertising industries would each build special applications to exchange information according to the agreed-upon document format), the language can be represented within the XML parlance and therefore interpretable by any XML-compliant interpreter. Theoretically, an XML interpreter would only require the particular language definition and an accompanying style guide for presenting the resulting document. This achieves significant economies of scale for development, since a company proposing a new interchange format would not need to build an interpreter to prove the concept.

Since these languages are created and published for the express purpose of industry adoption, the XML development community is determinedly open-standard and multi-platform, and XML languages are interpretable by design. With the vendors of traditional HTML browsers embracing this document standard, there is increasing use of XML as a base for defining new languages, particularly those that are transactional in nature.

While a new programming language could be defined to meet the aforementioned goals (interpretability, multi-platform, open source) without employing XML as the base language, the goals of the speech and XML communities clearly coincide in this case. Additionally, as service and content providers face pressure to broadcast content over an escalating number of distribution channels and devices, powerful *multimodal* authoring tools have been created specifically for XML languages. The cadre of engineers and managers at these companies cannot be distracted by supporting multiple, incompatible information infrastructures. The application is the data itself, and the web, phone, and Palm Pilot are simply vehicles for presenting it. If another presentation technology is required, such as WML and WAP, it must be consistent with and a natural extension of the infrastructure already in place: databases, servers, IP-based links, etc. Leveraging this familiarity with the network and tools leads to potential gains in production

[76] For a catalog of known XML specifications, see http://xml.org/xmlorg_registry/.

output and performance, as well as satisfying the core requirements of the Voice Web model. In short, XML purports to provide scale economies for multimodal distribution as well as language definition.

6.1.5. Utility

Perhaps the primary (yet often unstated) metric for gauging the value of a new speech language is *utility*: the applicability of the language to the tasks commonly encountered in the target market space. For example, a language that does not allow the creation arrays is ill-suited to implementing a matrix manipulation program. Whenever a new language is introduced, the general expectation is that the design process was informed, if not driven, by the failures of precursor languages - there is a parallel expectation that this new language would similarly refrain from introducing new restrictions. The application programmer is the final arbiter of a language's adoption, and any language that fails to meet the most basic criterion of utility will not even be considered notwithstanding the presence of other positive factors.

6.2. Development of VoiceXML & DialogML

6.2.1. Precursors

Traditionally, the ASR vendors were responsible for creating languages that not only provided an API to the engine itself, but also provided a coherent interface with all the key speech gateway subsystems - it was the responsibility of the gateway vendor to supply an execution environment consistent with that interface. As a result of this arrangement, there have been about as many speech languages as there have been ASR vendors. Even today's speech recognition vendors sport a multiplicity of incompatible dialog languages, grammar and synthesis formats, and development tools. Despite the hegemony of compiled languages, several companies began to experiment with interpretable languages in the early 1990s to shorten prototype development time and to encourage third-party organizations to build applications. There were two basic approaches to developing an interpretable language: create a new language (sometimes based on XML) or extend the already popular HTML.

The most notable of these early efforts was the PhoneWeb research project at AT&T Bell Laboratories, whose language was titled simply the Phone Markup Language (PML). After the spin-off of Lucent from

AT&T in 1996, some of the key PhoneWeb contributors stayed within the familiar AT&T-Lucent fold, while others left to create a similar project within Morotola. Both AT&T and Lucent continued to pursue PML, but the resulting languages, while sharing the same name, ended up with different syntaxes. Both Lucent-PML and AT&T-PML have been used to build call-center services and consumer telephone services, but they are available only on a limited number of platforms and the language definitions are tightly held.[77]

Given its prominence in the mobile productivity space, Motorola also embarked on a research program of developing a language that would support timely information access and interaction. Unlike AT&T's PML efforts several years earlier, Motorola's delayed entry into this space presented the opportunity to base its own language on the then-embryonic XML framework. In October 1998, Motorola announced its VoxML system: language specification, software development kit, Windows-compatible interpreter, ASR and TTS engines from third-party vendors. [78] In addition to the novelty of an XML-based speech language, Motorola furthered the case that a speech gateway need not be a custom hardware platform, but rather it could simply be a collection of internal and third party software on generally-available computer systems. While the VoxML system is still supported by Motorola, the notable absence of an open standard failed to coalesce the support of the entire speech community.

Concurrent to Motorola's development efforts, IBM began work on its Speech Markup Language (SpeechML) product in May 1998. SpeechML and VoxML are similar both in terms of the language definition and the use of general purpose systems, but IBM chose to support its internal ASR engine rather than integrate external elements. In addition to being targeted at traditional telephony applications, SpeechML was also intended to support automation of desktop applications.

Other efforts have focused not on creating a new speech language, but simply extending the already well-supported HyperText Markup Language. Netphonic (acquired by General Magic in 1998) had a product called Web-on-Call that employed proprietary extensions to HTML to provide telephone access to web services. Vocalis' SpeecHTML uses a subset of HTML to provide interactive voice services.[79] In addition to its Speech API language effort, Microsoft developed a Web Telephony Engine (WTE) that "renders"

---

[77] General background information on precursors to VoiceXML was gleaned from the AVIOS 2000 tutorial *Building Speech Recognition Telephony applications with Voice eXtensible Markup Language* (http://www.voicexml.org/avios2000_voicexml.pdf).

[78] See http://www.voxml.com.

[79] See http://www.speechtml.com.

standard HTML over the phone and uses a small number of proprietary extensions to support SIT functionality.[80]

6.2.2. Current Standardization Efforts

Of the two general approaches to developing a speech markup language-starting from "scratch" or extending HTML - VoiceXML clearly falls into the former camp, which clearly reveals the origins of this markup language. Recognizing the fundamental similarity of Lucent-PML, IBM-SpeechML, and Motorola-VoxML, and AT&T-PML, these four companies began to collaborate on a new, unified language specification called the *Voice eXtensible Markup Language* (VoiceXML). Shortly thereafter, these founding companies created the VoiceXML Forum, whose charter is to establish and promote this language as an open standard essential to making Internet content and information accessible via voice and phone.[81] A preliminary standard was released on 17 August 1999, and the first public release in March 2000.[82]

Concurrent to the formation of the VoiceXML Forum, the World Wide Web Consortium held a workshop on Voice Browsers in October 1998, bringing together various groups already involved in developing devices and browsers for accessing Web-based services by voice. Since these efforts involve either "subsetting" or extending several of the core W3C technologies (e.g. HTML, CSS), the W3C established the Voice Browser Activity and Working Group on 26 March 1999.[83] Another motivation for the W3C's longstanding tradition of advancing universal access: "To make the Web accessible to all by promoting technologies that take into account the vast differences in culture, education, ability, material resources, and physical limitations of users on all continents."[84]

---

[80] Excerpt from the Microsoft Developer Network: "The WTE uses only standard HTML elements. However, Microsoft has defined five items intended to optimize the use of HTML for aural browsers. Microsoft is working with the World Wide Web Consortium (W3C) to add the following items to the next version of HTML." See http://msdn.microsoft.com/library/psdk/webte/wteauthor1_7fub.htm.

[81] As of 7 March 2001, there were over 420 members of the VoiceXML Forum: http://www.voicexml.org/.

[82] VoiceXML Forum, *Voice extensible Markup Language Specification*, Version 1.00, 7 March 2000 (http://www.voicexml.org/specs/VoiceXML-100.pdf). Note that future references to this document will be abbreviated as *VoiceXML Specification*.

[83] "The lessons learned in designing for accessibility can be applied to the broader voice browsing marketplace, making it practical to author content that is accessible on a wide range of platforms, covering voice, visual displays and Braille." For an overview of the various Voice Web standardization efforts undertaken by the Voice Browser Working Group, see http://www.w3.org/Voice/.

[84] See http://www.w3.org/Consortium/.

The involvement of the W3C is significant in that it exerts great influence over the development and standardization of web technologies, a fact not overlooked by the VoiceXML Forum membership. On 7 March 2000, the VoiceXML Forum Version 1.0 Specification was submitted to and accepted by the W3C, and in May 2000, it further agreed to adopt VoiceXML as the basis for the future development of the W3C's own *Dialog Markup Language* (Dialog ML).[85] Whereas the VoiceXML Forum's scope was limited to the dialog language specification, DialogML is only one of several complementary standards undertaken by the W3C Voice Browser Working Group.[86]

Up until the blessing by the W3C, it was not clear whether the standards body would adopt an XML schema approach or simply extend HTML. The success of VoiceXML at the W3C was due not only to its technical merits, but as a result of the availability of prototypes from a significant number of companies. Even before the 1.0 release, the founding members and even several startups had already developed VoiceXML reference platforms. For example, Lucent' Speech Server, Motorola's VoxML Gateway, and Nuance's Voyager all supported Beta versions of the VoiceXML standard as early as January 2000, fully 3 months before the release of 1.0. These Beta products certainly did not showcase superior reliability or even complete compliance with the standard, but their early adoption efforts signaled a major vote of industry confidence. While there is no simple metric to gauge a language's success in meeting it's design goals, VoiceXML has certainly achieved sustainable momentum in terms of standardization and availability.

6.3. Overview of VoiceXML

Even though the forthcoming DialogML will effectively replace and eventually extend VoiceXML 1.00, the latter is the only language for which there is a detailed specification, so this paper considers VoiceXML as the relevant dialog language. However, both standards will be referenced in those cases where DialogML is anticipated to diverge from the VoiceXML specification.

Taking its cue from VoxML and the general direction of the Web industry, VoiceXML was specifically developed as an XML schema:

---

[85] Currently, there is no Working Draft of the DialogML specification, though a general Requirements Document is available for public review: http://www.w3.org/TR/voice-dialog-reqs/. Note that future references to this document will be abbreviated as *DialogML Requirements*.

[86] Two of these complementary standards were introduced in earlier chapters:

- Familiar client-server model of application flow and resource management
- Document interpretation/presentation and service logic are separable resources and can exist in different platforms. In other words, VoiceXML documents generally focus on presenting a single dialog (query and response interaction) and the remote server focuses on the higher-level business logic and back-end integration issues.
- The details of the specific hardware and software platform are abstracted away from the programmer
- Run-time interpretation allows quick prototyping of applications
- Re-use of existing content development and delivery tools already employed for the HTML-web
- Open standard promotes service portability across implementation platforms

While a meticulous introduction to the finer points of the VoiceXML lexicon is beyond the scope of this paper, there are several aspects of the language that impact its ability to support the Voice Web architecture.

6.3.1. Speech Gateway Control

Typical of an XML schema, VoiceXML operates at quite a distance from the underlying hardware. Just as an HTML programmer is unconcerned with the technological alchemy by which a graphic file is displayed on the user's screen, the VoiceXML programmer does not have to manually control the various elements of the speech gateway. This *declarative* abstraction is sufficiently precise to support most simple dialogs and interactions; however, in those cases where finer-grained control or additional functionality is desired, the programmer must necessarily sacrifice code portability.

Input Technology. While VoiceXML provides standard support for DTMF input, the specification spends considerable effort in creating programming structures for general-purpose ASR, relying extensively on programmer-supplied grammars to guide the recognition process. There are various elements that are implied or left undefined in the VoiceXML specification:

Speech Recognition Grammar Specification and Speech Synthesis Markup Language.

- *Word-spotting* and *speaker independence*. While neither of these terms appears anywhere in the VoiceXML and DialogML documentation, the examples and discussion indisputably imply that these are intended as the default mode of speech input.[87]

- *Natural language processing*. Since the VoiceXML browser (as defined) does not maintain a state or history of dialogs, non-standard mechanisms must be supplied to support natural language processing. While there is always debate in the application community over the utility of natural language dialogs in a mass market application, the W3C Voice Browser Working Group has created a parallel standardization effort: *Natural Language Semantics Markup Language.*[88]

- *Speaker identification and verification.* While these functions are commonly found in speech gateway systems, they are simply not included in the VoiceXML standard. The DialogML Requirements document lists speaker verification as a "should have."[89]

- *Barge-in*. While this is not a required functionality of the implementation platform, VoiceXML provides explicit support for this feature;[90] again, DialogML lists this feature as a "should have."[91] No mention is made of whether the barge-in mode is energy-based or recognition-based, but presumably the programmer would be able to select this mode through a platform-specific procedure.

- *N-best matching*. As already described in the Input Technologies chapter, N-best matching extends the results of the ASR engine to include an array of values and match scores, rather than a single value-score pair. Neither the VoiceXML nor DialogML specification makes any mention of this functionality.

- *Grammars*. Recognizing the importance of dynamic dialog generation, VoiceXML does not require the pre-compilation of grammars; additionally, VoiceXML regards the grammar as a

---

[87] This does not disqualify speaker-dependent implementations, but there is no mechanism in the standard for conveying user-specific training data to the ASR engine.
[88] The Natural Language Semantics Markup Language (NLSML) was published as a Working Draft on 20 November 2000, and no Recommendation date has been set. It is unclear whether DialogML and NLSML will merge or remain distinct languages in the long-term. See http://www.w3.org/TR/nl-spec/.
[89] Section 4.3 of *DialogML Requirements*.
[90] Section 13.5 of *VoiceXML Specification* describes the `bargein` attribute of the `prompt` tag; however, note that barge-in is not listed as a required platform feature in Section 2.5.
[91] Section 2.7 of *DialogML Requirements*.

general Web resource, so it can be flexibly located anywhere on the Internet.[92] While VoiceXML does not specify a grammar format, the W3C's Speech Recognition Grammar Specification addresses this oversight.[93] Recalling the difficulty in creating and testing grammars, the standard provides several *built-in* grammars to implement the most frequently-encountered dialogs, such as "currency" and "date."[94]

- *Multi-Lingual*. For those speech gateway platforms that support multiple languages, both VoiceXML and DialogML provide support for designating the active ASR engine.[95]

- *Grammar Scoping*. Recalling the discussion in the Input Technology Chapter (page 59), grammar scoping not only allows the programmer to flexibly define grammars both within and across dialogs, it offloads much of the management process for activating and deactivating grammars as appropriate. VoiceXML provides various levels of scoping within a single document, but only one scope (*application* scope) that reaches across page boundaries;[96] judging from the requirements document, DialogML potentially adds one more multi-document level: *session* scope.[97] While the finer points of grammar scoping appear irrelevant in the context of a strategic analysis, they play a surprisingly important role in the creation of the Voice Web.

Interface Technology. While the telephony interface is often considered simply as the means to the ends of speech recognition, VoiceXML provides limited support for call handling:

---

[92] Excerpt from Section 2.5 of *VoiceXML Specification:* "[The implementation platform] must be able to receive speech recognition grammar data dynamically. Some VoiceXML elements contain speech grammar data; others refer to speech grammar data through a URI."

[93] Appendix D of the *VoiceXML Specification* suggests the adoption of Sun's Java Speech Grammar Format. As already discussed in the Input Technologies chapter, the W3C's own SRGS is based upon this JSGF format (see page 65 for the full discussion).

[94] Section 14.1.1 of the *VoiceXML Specification* describes the usage and types of built-in grammars, including Boolean, date, digits, currency, number, phone, and time.

[95] Section 5 of the *VoiceXML Specification* briefly mentions the `lang` attribute of the `voicexml` tag. Section 4.2 of the W3C SRGS document describes a similar `lang` attribute for the root-level `grammar` tag, and section 6.5 argues for the provision of phonemic pronunciations, particularly relevant for those words for which the written form does not indicate the correct pronunciation.

[96] For a discussion of grammar scoping and activation rules, see Sections 10.3 and 10.4 of the *VoiceXML Specification.*

[97] Section 3.5 of *DialogML Requirements.*

- *Transfers*. The user's incoming call can either be bridged with a third line (i.e. bridging transfer) or completely offloaded to the third party (blind transfer).[98]

- *Recording*. The caller's speech fragment can be directed to an audio file, either in addition to or in place of speech recognition processing.[99]

- *Call Management*. For the purposes of identifying the caller and potentially tailoring the call flow in response, a variety of standard session parameters can be recovered for the current call (e.g. ANI and DNIS), provided that they are available from the telephony service provider.[100]

Output Technology. VoiceML generally enables the unique output attributes of the Voice Web without sacrificing support for general SIT application, but there are notable omissions as well.

- *Remote audio*. Just like a typical HTML page with links to other pages and images, a VoiceXML document can flexibly include any resource accessible over the Internet, including audio files. In the Voice Web Model, the consumer (not the portal) drives the ad hoc, instantaneous "relationship" between the speech gateway and the application code, so remote access to audio is an indispensable feature.[101]

- *Audio formats*. The specification does not require provision for any particular audio type, but it suggests in an appendix that the static/8-kHz/8-bit/mono format of the PSTN be natively supported.[102] Apart from only the briefest mention in the DialogML materials, there is no provision for streaming audio, regardless of format.[103]

---

[98] Section 14.7 of the *VoiceXML Specification* describes the operation of the `transfer` tag, and Section 2.8 of the *DialogML Requirements* lists call transfer as a "should have."
[99] Section 14.6 of the *VoiceXML Specification* describes the `record` tag.
[100] Section 9.4 of the *VoiceXML Specification* lists the standard session variables. For a review of the kinds of identification data associated with an incoming call, see the footnote on page 37 of this paper.
[101] Nearly every aspect of the VoiceXML document can be externally referenced through a universal resource identifier (URI). The most familiar URI schemes are the mnemonic HTML website addresses (e.g. http://www.ibm.com), but they can flexibly refer to any resource on the Internet. A partial listing of URI schemes can be found at http://www.w3.org/Addressing/schemes, and the official URI syntax is defined in RFC 2396 (http://www.ietf.org/rfc/rfc2396.txt).
[102] The four audio formats suggested in Appendix E of the *VoiceXML Specification* are the same except for the header and encoding types (see the discussion on page 40 for a review of encoding formats).
[103] Streaming audio is listed as a "must have" in Section 3.9 of the *DialogML Requirements*, but relegates the task to the Speech Synthesis Markup Language. However, this directly contradicts Section 2.10 of the SSML Specification, which explicitly discourages the use of SSML's `audio` tag as a general output mechanism and makes no mention of streaming audio.

- *TTS markup*. VoiceXML provides a handful of elements for guiding the TTS engine, but the W3C's Speech Synthesis Markup Language provides robust text annotation, including support for international languages and phonetic alphabets.[104]

- *Long-play support*. Unlike the typically short audio prompts, certain applications of long-play audio feed require internal navigation cues, such as the offset in bytes or seconds from the beginning of the file. For example, if a barge-in unintentionally occurs during the playback of hour-long feed, then the user should be able resume where the interruption occurred rather than be forced to start from the beginning. Another example is an audiobook application: the user may want to fast-forward through the boring parts or to bookmark a passage to resume playback later. Since neither VoiceXML nor DialogML provide granularity beyond the file boundary, there is no long-play support other than trivial playback.

6.3.2. Extensibility

In addition to providing a consistent service interface the speech gate infrastructure, the designers of VoiceXML recognized that they could not anticipate nor include every possible feature desired by speech programmers. In addition, the benefit of hindsight afforded by HTML also suggested that the strict client-server, markup language paradigm might not always work well within a telephony environment.

- *Executable content*. The canonical view of markup languages is that they should only contain declarative (presentation-layer) elements. However, the linear dimensionality of the telephony voice user interface,[105] coupled with the expectation for high-frequency/low-latency computer-user interaction, demands the inclusion of certain executable directives in the lexicon. At the risk

[104] VoiceXML essentially adopts a subset of Sun's Java Speech Markup Format. As already discussed in the Output Technologies chapter, the W3C's own SSML is based upon this JSML format (see page 79 for the full discussion). For issues of multinational support within the *SSML Specification*, refer to Section 2.2 for a description of the `lang` attribute of the root `speak` tag, and Section 2.5 for a discussion of the `phoneme` tag.

[105] Unlike an HTML document, the order of execution matters on a VoiceXML page. For instance, an HTML page that refers to a variety of graphics files can load them in any order, even in parallel, without interfering with the programmer's intent. In contrast, the one-dimensional, linear aspect of the audio-only interface requires that the content files be played in a particular sequence, e.g. "high temperature of" + "fifty degrees" is very different from "fifty degrees" + "high temperature of".

of evolving into a procedural language, VoiceXML provides a small set of executable elements to specify call flow logic locally (on the interpreter): if-then, goto, reprompt, exit, etc.[106]

- *Scripts*. A certain type of executable content in VoiceXML that bears further examination is the script. In addition to the handful of procedural elements native to VoiceXML, it also supports the inclusion of general procedural code (or scripts) for execution on the speech gateway. Leveraging the standardization efforts of the HTML Web, VoiceXML scripts must be consistent with the European Computer Manufacturers Association (ECMA) specification: *ECMAScript*.[107]

- *Platform-specific procedures*. The abundance of proprietary plug-ins to HTML browsers (e.g. Shockwave, RealPlayer, Acrobat Reader) provides ample evidence that no single language can do everything. Recognizing the inevitable and legitimate need to extend the functionality of the language, VoiceXML includes a special markup element[108] through which gateway vendors can implement nonstandard features, such as speaker verification and reusable dialog components.[109]

6.3.3. Multimodality

Even though this paper focuses ostensibly on the Voice Web, any strategic analysis should not fail to consider the role of multimodal applications and devices in shaping the overall Web industry. More specifically, VoiceXML's relevance may be determined, in part, by its ability to evolve into or coexist alongside a true multimodal language. Unfortunately, the various applications of the term *multimodality* obscures it's meaning in the context of Web publishing. If it is intended to describe any process that facilitates the creation of documents in a variety of markup languages, particularly from a single document standard, then XML-based publication is inherently multimodal. For example, if a single

---

[106] Section 19 of the *VoiceXML Specification* enumerates the executable tags.

[107] ECMA (http://www.ecma.ch/) is an international standards body that specializes in the fields of information technology and telecommunications. The ECMAScript specification can be downloaded at ftp://ftp.ecma.ch/ecma-st/Ecma-262.pdf. In the course of the researching this industry, the author noted that the term *JavaScript* is often used interchangeably with ECMAScript; though technically incorrect, the distinction is negligible in practice.

[108] Section 14.5 of the *VoiceXML Specification* describes the use of the `object` tag.

[109] Recalling the airport sample application on page 56, virtually all ASR vendors provide reusable dialog components to automate the logic and grammar creation process for common tasks, some of which are co-opted by VoiceXML's own built-in grammars, e.g. currency, date, etc. However, these RDCs can also incorporate extensive call flow logic, capably of implementing entire applications, e.g. automated airline reservation. For more information, the reader can refer to the RDC pages of the following two ASR vendors: SpeechWorks DialogModules (http://www.speechworks.com/productsservices/products/speechworks6/overview/dialog_modules.cfm) and Nuance SpeechObjects (http://www.nuance.com/products/speechobjects.html).

database can be parsed to produce both VoiceXML and WML, then multimodality is satisfied in that there are two interfaces (visual and aural) to a single repository of business logic and data; however, these interfaces are not coordinated or synchronized to work in conjunction. Unlike the rich multimedia experience afforded by the HTML Web, VoiceXML and WML were not designed to work together to enable a single, multimodal interface. In some sense, the concept of multimodal publication is the converse of the multimodal interface:

- *Multimodal publication*: tools and practices that facilitate the creation of multiple interfaces from a single markup schema
- *Multimodal interface*: a language or a family of compatible languages that facilitate the creation of a single interface

From this perspective, VoiceXML merits the moniker of multimodal publication by virtue of its foundation in XML, but no claims of multimodal interface support were expressed or implied in the 1.0 specification. VoiceXML's designers anticipated the need to look beyond audio-only mobile interfaces, prompting the joint workshop between the W3C and the WAP Forum.[110] Even a cursory view of the attendees' position papers demonstrates the multiplicity of architectures for delivering multimedia applications. The result of this meeting was the *Hong Kong Manifesto,*[111] which outlined the need to create a new working group to synthesize the concerns of the attendees and to create actual use scenarios to drive the next phase of the standards process. Within the W3C, the task has been taken up by the Multimodal Systems subgroup of the Voice Browser Activity, which has gone as far as releasing a requirements list for multimodal markup languages.[112]

In addition to the work of this newly formed Multimodal Systems activity, the W3C's other efforts in the User Interface Domain may bear upon the evolution of the mobile Web's multimodal interface. The Device Independence Activity's goal is to discover mechanisms that allow a single markup page to viewed on any device; however, this is more of an attempt to cater to the lowest common denominator of web publishing, rather than the creation of the most appropriate multimodal interface for mobile devices.[113] The Synchronized Multimedia Activity has yielded an XML language that tackles the issue of

[110] W3C/WAP Workshop: the Multimodal Web. Shangri-La Hotel, Hong Kong, 5-6 September 2000 (http://www.w3.org/2000/09/Papers/Agenda.html).
[111] See http://www.w3.org/2000/09/Papers/Meeting/recommendations.html
[112] *Multimodal Requirements for Voice Markup Languages*, W3C Working Draft, 10 July 2000 (http://www.w3.org/TR/multimodal-reqs).
[113] See http://www.w3.org/2001/di/.

temporal behavior in a single multimedia presentation (e.g. video stream annotated with hyperlinks), but it is not clear whether this work applies to more general task of Web browsing.[114]

6.3.4. Browser Functionality

To be sure, the markup language is the foundation upon which the interface is crafted, but it does not circumscribe nor enable all facets of the user experience. In addition to performing the "commodity" function of rendering the markup code, the browser itself enables certain elements of the user interface. Consider the following screenshot of an HTML browsing session:

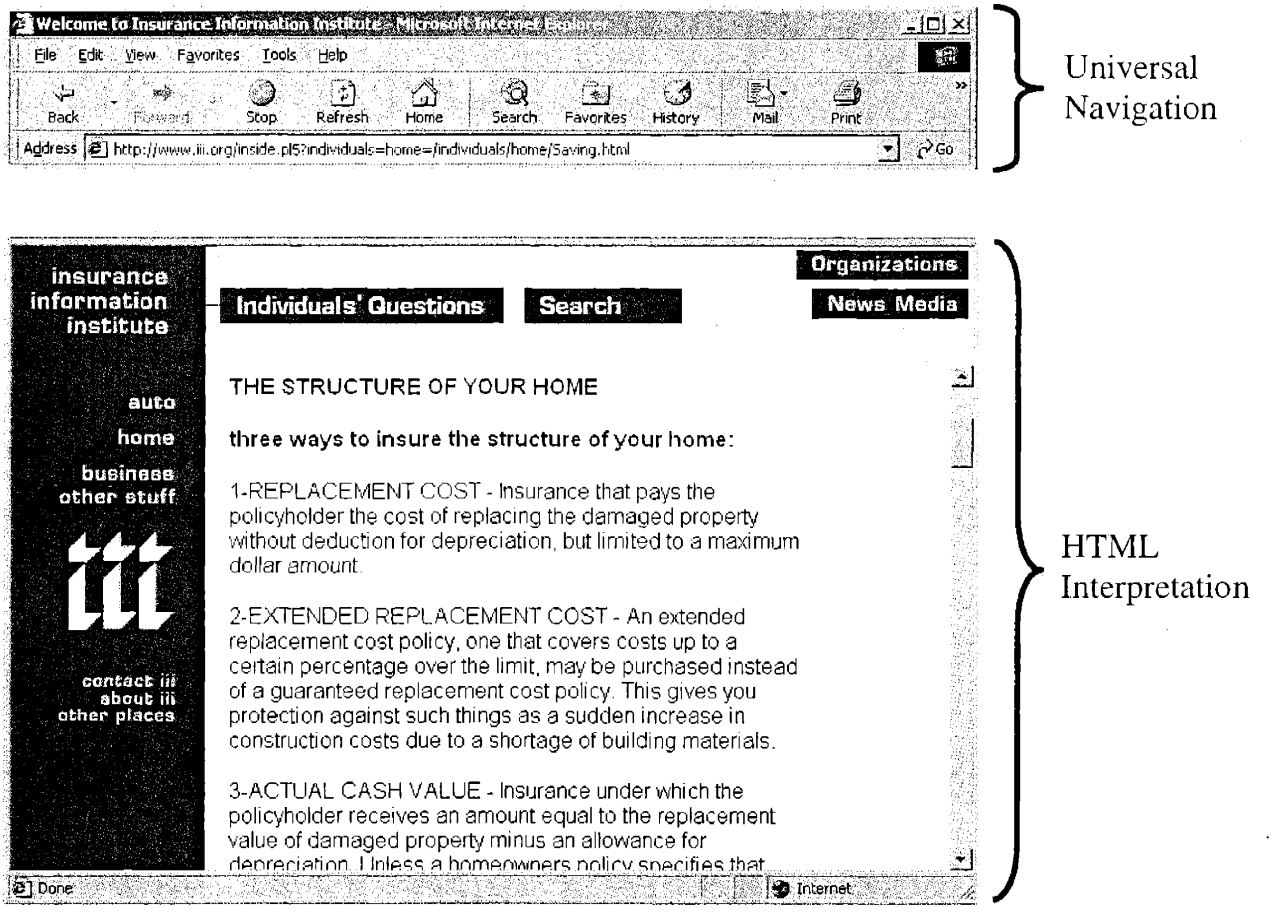

Figure 28. Functions of HTML Browser (Universal Navigation and HTML Interpretation)

The lower portion of Figure 28 does indeed represent the core HTML rendering component, but the upper portion provides *universal navigation* functionality that exists independently of and across Web

[114] See http://www.w3.org/AudioVideo/.

destinations, e.g. stop, back, refresh, home, website specification, etc. As will be demonstrated in the Structural Analysis, the availability and standardization of this short list of commands play a key role in enabling the universal and democratic access characteristic of the Web. While VoiceXML makes no attempt to provide these navigation constructs, DialogML hints at this type of browser-level functionality:

> "The [DialogML] markup language specifies a set of meta-command functions which are available in all dialog states; for example, *repeat*, *cancel*, *quit*, *operator*, etc. The precise set of meta-commands will be coordinated with the Telephony Speech Standards Committee. The markup language should specify how the scope of meta-commands like *cancel* is resolved."[115,116]

Interestingly, there is no browser-level interface standard for the HTML Web,[117] but the force of convention has engendered a form of *de facto* standard respected by most vendors, even if they sometimes employ different names, e.g. Internet Explorer "Refresh" and Navigator "Reload" are functionally equivalent.

6.4. Summary

While this chapter has largely focused on the enabling features of the language itself, the VoiceXML Interpreter forms the foundation of the Execution Environment and the focus of the integrative energies of the gateway architects. Like any other element within the speech gateway, it must satisfy the run-time performance and scalability parameters established for the entire system.

If the number of commercial VoiceXML platforms is any indication, then the standard has reached "escape velocity" from its origins in theory and committee. Whereas other languages have failed to capture the attention of the speech community, VoiceXML (and its eventual replacement, DialogML) alone satisfies the core requirements: portable, open, interpretable, and useful. To be sure, the markup language and client-server approach to dialog creation is a significant departure from conventional speech programming, but it suits well the dynamic content and logic typical of the Web.

---

[115] Section 2.6 of the *DialogML Requirements* lists meta-commands as a "should have."

[116] Despite the author's best efforts and repeated requests, there is no evidence of the *Telephony Speech Standards Committee* within the W3C or the VoiceXML Forum. The closest reference that the author can uncover is the *Telephone Speech Standards Committee* whose website has not been updated for almost 2 years (http://www.delphi.com/speechreco). Some older information can be found at: http://www.speechcentral.com/content/tssc.html.

[117] The closet thing to a browser standard is the W3C User Agent Accessibility Guidelines (http://www.w3.org/TR/UAAG10/).

At this point, each element of the speech gateway has been detailed: interfaces, input technology, output technology, and execution environment. However, there are additional components outside the gateway but within the speech interactive telephony architecture that are required to fulfill the real-time and intuitive access characteristics of the Voice Web. These *enhanced network services* will be the subject of the following chapter.

Chapter 7. ENHANCED NETWORK SERVICES

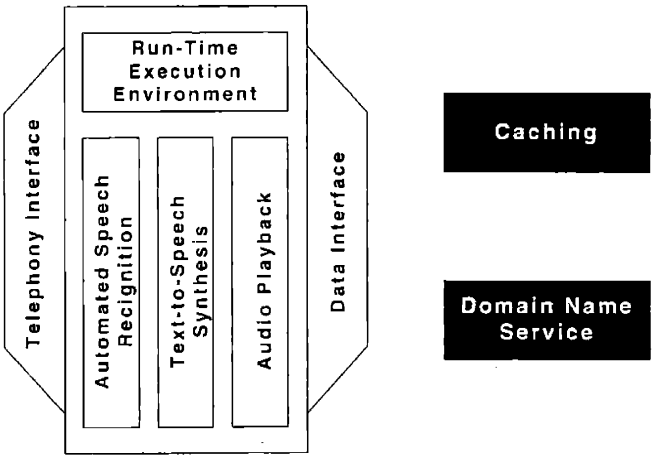

Figure 29. Speech Gateway (Currently on Enhanced Network Services)

In Figure 5 on page 28, the Reference Model enumerates the various components for program interaction: interface, execution, application, and the links between each of these elements. While that model provides a framework for understanding the configuration of the SIT architecture (e.g. service provider-side interpretation), it cannot capture all facets of the Voice Web experience. In particular, noticeably absent from the model is the requirement for *real-time* and *intuitive* access to audio content, both of which are essential for creating a true Voice Web (but are not necessary unique to it).

In the case of real-time interaction, HTML users already experience the frustration of engaging real-time media streams (e.g. audio and/or video) over a variable latency, public data network. In addition, operators of popular destination sites regularly struggle with balancing and servicing the traffic load on their network with a lower bound on service quality. The challenge of delivering real-time audio for a Voice Web application is essentially identical to that of publishing rich media content over the Internet generally. Voice Web users may very well prove to be even less tolerant since they have no visual feedback to occupy the "downtime" or "buffering," coupled with the already high expectations for traditional telephony service. This chapter will explore how *caching technologies* fortify the application service provider's ability to deliver rich media content in a timely and consistent manner.

In consideration of the need for intuitive, democratic access to Web destinations, the *domain name service* (DNS) provides a mechanism that replaces complex and unintuitive IP addresses with (hopefully) mnemonic names. While the DNS works quietly behind the scenes converting web addresses into their IP

equivalents, the importance of this "commodity" translation service should not be underestimated: ad hoc, intuitive access would be fairly impacted by the elimination of addresses based on familiar brands (e.g. www.mcdonalds.com) and categories (e.g. www.news.com). While the DNS has proved sufficiently flexible in extending support to a variety of applications (e.g. telnet, ftp, http, email), the Voice Web's unique audio-only interface may ultimately require a modification to or replacement for the existing DNS system.

7.1. Caching Technology

Since the Voice Web model actually encourages content to be distributed all over the Web, the quality of audio playback depends not only on the computational horsepower of the output system, but also on the latency and throughput characteristics of the Internet. Even casual users of web-based audio technologies are familiar with the standard startup buffering delay to guard against missing, late, or corrupted audio, and the inevitable corruptions in the audio signal that occur anyway - it's not uncommon for the session to simply seize up entirely due to network congestion. While Internet consumers are accustomed to shoddy audio performance on the computer, they will not likely be as forgiving when accessing information over the telephone: imagine a user squandering her precious cellular airtime minutes waiting for an audio feed to resume at some unforeseen point in the future. On the computer, people have obvious visual cues when the stream is experiencing transmission problems, and the audio is typically a secondary task providing nonessential background music. On the phone, the only hint that the signal is degrading is to suffer through it, and the audio is the primary task providing the entire user experience.[118]

The currently competing design goals of flexible content distribution and high throughput to support audio playback must be reconciled to enable a Voice Web configuration. This is not a performance issue unique to the Voice Web, since Internet telephony and even standard content providers want to minimize the delays associated with transmission, but may make or break the service in terms of market acceptance. Dedicated hosting and serving equipment at the remote content site do indeed reduce the onsite latency, but they do nothing to mitigate the effects of a packet-switched best-effort delivery system. Short of

[118] There are many reasons why commercial Internet telephony has not significantly penetrated the traditional telephony market, but the main complaint from the users is that the audio signal does not meet their quality expectations. One could extrapolate that SIT application users will be similarly intolerant of deficiencies in the audio playback.

contracting for QoS expensive connections to each and every audio site on the Internet, the most effective solution is to incorporate *caching* technology.[119]

Caching is the replication of **audio data** (or more generally, any data) and **serving functionality** for the purposes of decreasing latency between the server and client processes. Most web browsers directly support a rudimentary form of caching by locally storing recently accessed objects: the VoiceXML specification itself allows the programmer to establish caching policies for each object.[120] External caching servers extend this basic replication concept to include more complex policies and data types. Caching servers can either passively replicate content by observation of the clickstream, or they can proactively grab content from remote sources based on location, update interval, and other parameters. The proactive process provides better performance if the content universe accessible through a particular gateway is "closed" (e.g. TellMe), and a hybrid approach works well if the voice service provider additionally allows access to general audio content on the Web. In either case, the uncertain path between the remote content server and the local audio subsystem is replaced with a completely local playback loop: separated only by the VSP intranet, the audio player (client) and server can operate at a much higher throughput and with a minimum of delay.

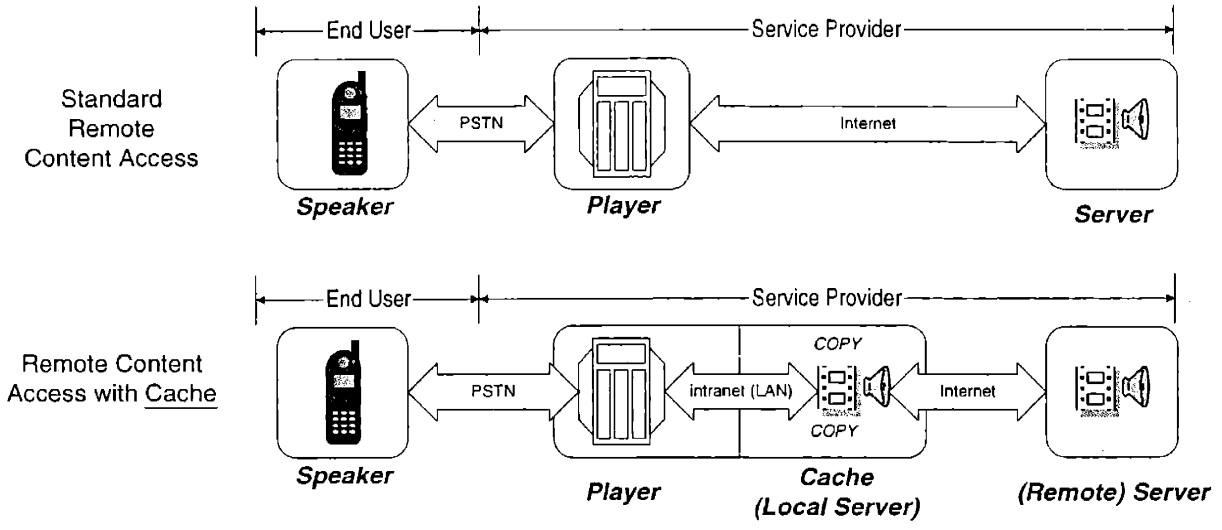

Figure 30. Caching Architecture

[119] While caching systems can dramatically affect the audio performance of a speech gateway, it is technically not considered to be a gateway component *per se*. A cache does not endow the system with any additional functionality; rather it mimics the functionality of a remote system in a manner completely transparent to the gateway.
[120] Section 12.2 of *VoiceXML Specification* describes the *fast* and *safe* caching policies.

Caching servers certainly increase the operational cost, but they are necessary to delivering audio content consistent with telephone expectations. Another benefit is the reduced reliance on the performance of the content provider's own audio servers, a fact much-appreciated by VSPs and ASPs alike. The content organization need only provide infrequent and low-bandwidth access to their servers, since the local cache will assume the responsibility for management of and distribution to potentially thousands of callers. For those rare services delivered over non-standard networks (e.g. National Public Radio's *Earth & Sky* is distributed on a CD with one month of content), the cache can also provide a normalizing function: pure HTTP access to the cache can abstract away the idiosyncratic details of each particular link technology. So without incurring any additional production or distribution costs, the content agency has now gained access to an entirely new and sizeable market: people who own phones. While the VSP does indeed assume additional serving costs and responsibilities, it has further ensured the audio playback performance demanded by the callers.

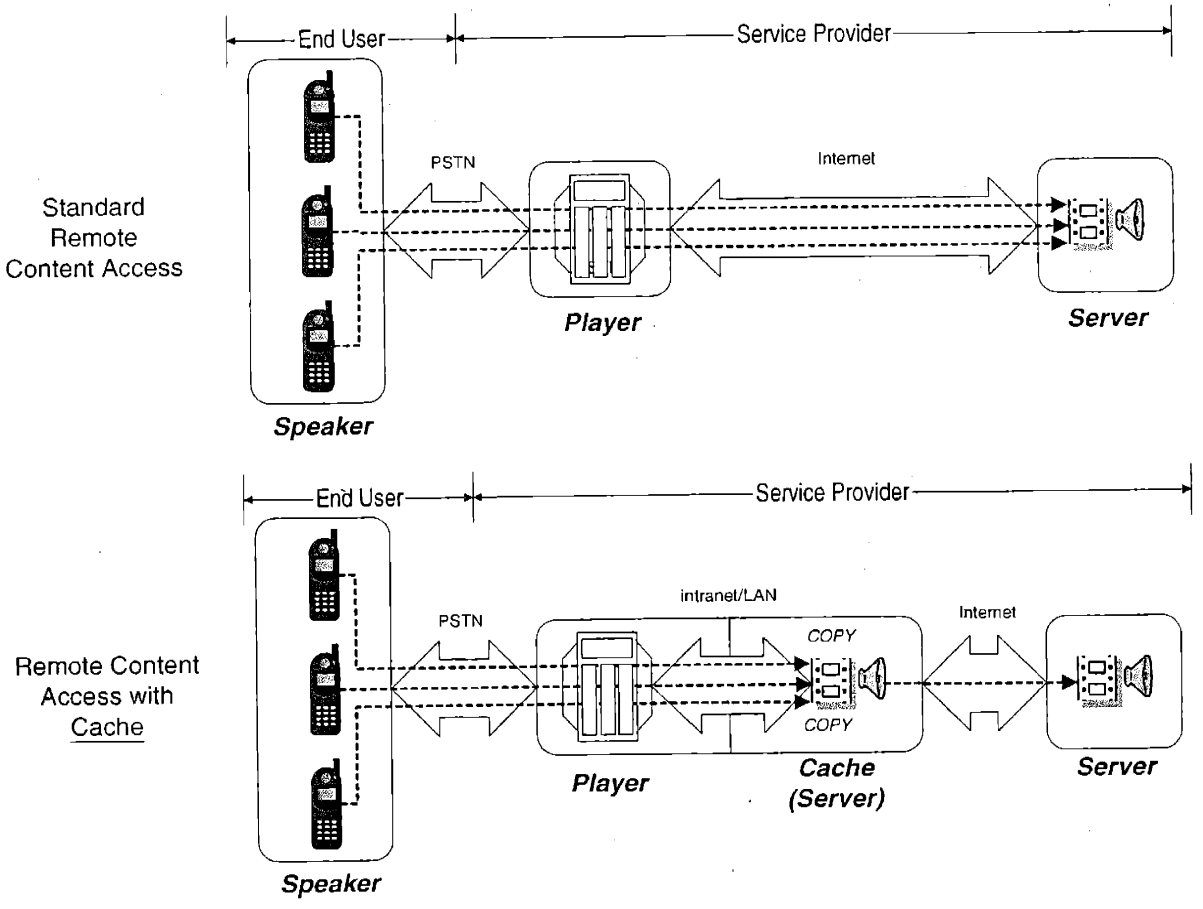

Figure 31. Bandwidth Savings Afforded by Caching Architecture

So far we have described the case in which the audio file is available for offline download from a remote content server, and that the player and server communication occurs over the phonecaster's local network. But if the file is already available in streaming format, one might imagine that it is a simple matter to cache the incoming audio streaming for future playback. A little examination reveals that since the audio player-and-server combo are proprietary elements from a single vendor, the ability to cache these streams is limited. The streaming technology was designed with broadcasting in mind, so the technology providers did not want to supply a means by which to make unauthorized copies of content.[121] The prevalence of software piracy created legitimate fears that once content found its way onto the Internet, unauthorized redistribution would occur, thereby eroding the financial opportunity for the production company and legitimate distributors.[122] Yet it is this very process of copying and caching that is required in order to ensure quality of service for phonecasting consumers.

In recognition of the apparent conflict between streaming and caching, a new generation of hardware has emerged called variously *proxy servers* or *content delivery servers.* In addition to the typical benefits of caching -bandwidth reduction on back-end and decreased reliance on Internet and remote servers-these proxy servers have worked closely with the major media streaming vendors to permit both real-time and asynchronous caching of the audio stream in its native format. In order to allay the licensing fears of content providers, these proxy systems incorporate *digital rights management* systems that inform the origin server of each stream request or follow a prescribed copyright policy, e.g. if only certain members of a service can download music, the origin server can authenticate the user identify and then choose to send the signal to release the stream at the proxy. There are only a handful of streaming-capable proxy vendors, but the important point is that they do indeed exist and can reconcile the needs of both streaming and caching.[123]

Implicit in the discussion above is that the audio files are all *pre-recorded*: they are available for complete download prior to the first user access. A natural question is how caching technology is capable of

---

[121] Opportunity for reader participation: The next time you find yourself about to click on a link that leads to a streaming audio file, try to save the file to your local hard drive (e.g. in Internet Explorer, 'right-click' the mouse and select 'Save As...')-you won't be able to do it. The audio file exists as a file only on the remote server (and even this condition does not hold for live content): only the player can intercept and output this proprietary protocol stream and encoding.

[122] The case of Napster (www.napster.com), the online music sharing community, provides ample evidence that this fear of unauthorized copying is well-grounded.

[123] Examples include: CacheFlow's CacheFlow (http://www.cacheflow.com), Inktomi's Content Delivery Suite (http://www.inktomi.com/products/network/products/cds.html), and Lucent's imminet (http://www.lucent.com/serviceprovider/imminet/). If the reader is willing to look past the obvious marketing bias, CacheFlow provides a good source for pursuing more information about content delivery servers: http://www.cacheflow.com/technology/whitepapers/.

preserving low-latency distribution for *live* audio content. The simple answer is that it cannot, since the fundamental limitations of the public Internet cannot be overcome by a local caching element. While the caching servers play an integral role in managing and serving the live streams across the VSP's intranet, other than delaying the entire broadcast for all users to permit some amount of buffering, there is little that the caching elements alone can do.

In these cases, the link itself between the ASP and the VSP needs to be modified. Other than the public Internet, there are a variety of communications technologies that afford a higher level of latency performance: point-to-point frame relay, virtual circuits over private ATM or IP networks, satellite dish relay, etc. Since these expensive fixed-links work outside of the normal cache-and-playback architecture, the VSP must select judiciously which ASPs to accommodate in this custom manner.

A more reasonable approach to the latency concern for live streams is to create "islands of content" that are connected by high-speed links and that are strategically located near the target audience and Tier-1 Internet exchanges. For example, if a high volume website like Yahoo! wanted to improve overall responsiveness, it would contract with a firm like Akamai (and incidentally, they did) to dynamically and proactively replicate Yahoo!'s site throughout Akaimai's FreeFlow network. This typically requires the content provider to adopt proprietary systems that communicate update and "freshness" information to the FreeFlow network (e.g. "Akamaize" the site), but the tradeoff is noticeably superior streaming content delivery over the public Internet. The philosophy is basically the following: *the Internet is a tough place for streaming, so if you can't locally cache the stream, at least keep the trip as short as possible.*

Yet another implication of live content is the efficiency of the computational load on the speech gateway. Every stream, whether live or pre-recorded, must undergo a serial process of decompression, reassembly and delivery within the player module (a process only complicated by additional requirement of real-time for live streams). Since a live stream exists independently of the users (no matter when a caller subscribes to the live stream, the content is "picked up" in mid-stream), there should be an economy of scope that centralizes the processes of decompression and reassembly. In other words, the cache (as a central component) should process all live streams into a format suitable for the particular speech gateway, and then redistribute that transcoded stream to its subscribers. In Internet parlance, the audio feed should be *multicasted*, not *multiply-unicasted* to the relevant callers. The closest analog in the telephony space is the conference call: all callers are multiplexed onto a single broadcast signal -callers don't make separate calls to each of the conference participants. Fortunately, the proxy streaming systems described above

typically offer multicasting support as a matter of course, since the decompression and reassembly costs should not have to be borne by each instance of the live session. Multicasting is critical to supporting not only live broadcasts, but also live chat, interactive multi-player games, and other applications that involve large groups of callers. An illustration of multicasting would appear exactly like the Figure 31, with the stipulation that the streams are inextricably linked and generated in real-time.

7.2. Domain Name Service

An indispensable aspect of every HTML browser interface is the ability to enter a (hopefully) mnemonic sequence of characters in order to access any site on the HTML Web. The presence of this feature is the democratizing force of the Web, since site-to-site excursions are not dependent on the links available on the default or initial page. In the case of the HTML Web, this sequence of characters is most commonly known as the web address, but is more formally known as a *uniform resource identifier* (URI). While the HTML address is only one of many possible URI schemes, it is by far the most common.[124] Once an accredited Registrar accepts a domain name, it is propagated to the authoritative and regional Domain Name Servers appropriate for that top-level domain. This global network of DNS servers is responsible for resolving each host name into a numeric IP address, e.g. "hotwired.com" resolves to "204.62.131.129." The routing elements of the Internet understands only these numeric IP addresses, so domain name resolution must be completed before the destination can be successfully accessed. The browser does not require the use of domain names, but it would be very cumbersome to advertise and remember arbitrary numeric sequences, particularly for branded sites.

There is a uniform IP addressing space, but there are a variety of services that operate on top of the IP layer, e.g. http. The URI convention was specifically designed to accommodate new services that rely on IP: each new protocol or document format need not require its own DNS service. Some services have their own URI schemes (e.g. 'mailto') while others simply borrow the existing HTML convention (e.g. WAP addresses utilize the familiar 'http' scheme). However divergent the actual services delivered through URIs and the DNS system, the tie that binds them together is the assumption of *text addressing*. For example, WML and HTML sites rely on entirely different document and protocol technologies to implement their respective services, but both rely on text addressing: e.g. http://wap.tdwaterhouse.com and http://www.tdwaterhouse.com. There is no denying that entering in long addresses through the WAP

[124] See http://www.w3.org/Addressing/schemes for a partial listing of URI schemes.

kaypad interface is cumbersome, but the resulting addresses is structured from the same subset of about 60 ASCII characters. [125]

Since the Voice Web, like any other network of peer sites, fundamentally relies on ad hoc and democratic access to any site connected to the Internet, there must be a corresponding mechanism of specifying addresses in an intuitive manner. There is no technological reason why callers can't simply speak each character of the domain name (nor each digit of the IP address), but human factors considerations demand a DNS-like service for the Voice Web. Unlike WAP, however, the Voice Web system must account for challenges of speech-directed URL specification. In other words, the Voice Web violates the common assumption of text addressing. The most obvious challenge is to match a single utterance against a library of potentially hundreds of thousands and to disambiguate the correct selection from other likely matches - there is a real concern of scalability over the long-term. In addition, consider the difficulty in distinguishing short names with similar pronunciations (bee.com vs. pea.com), true homonyms (carrot.com vs. karot.com, I.com vs. eye.com), etc. From a regulatory perspective, the final solution must address the potential trademarking contention caused by homonyms and preserve the vendor independence of the DNS system. The Structural Analysis section suggests several solutions to overcoming these technological, regulatory, and human factors constraints.

7.3. Summary

In conjunction with the elements of the speech gateway, caching servers and the domain name service form the complete picture of the speech-interactive telephony architecture. While caching and intuitive access are not absolutely essential to creating a web architecture, quality of service and experience certainly matter if the Voice Web model is to provide a meaningful space for application development. The next and final chapter of this Technology Primer brings these elements together to foster an understanding of the systemic operation of the SIT framework.

[125] The official URI syntax is defined in RFC 2396. See ftp://ftp.isi.edu/in-notes/rfc2396.txt.

Chapter 8. SPEECH INTERACTIVE TELEPHONY ARCHITECTURE REVISITED

In order to reinforce and integrate the preceding discussion, as well as prepare the reader for the Strategic Analysis to follow, this section will trace the functionality and data flow through the architecture for a sample interaction with a caller.

8.1. Case of VirtualPortal.com

Apart from the speech gateway, the SIT architecture closely resembles a traditional HTML (or WML) network, so many of the same service provider elements can be used to deploy either service: databases, routers, servers, intranets, etc. The following diagram illustrates the architecture for a fictitious *audio portal*[126] company named VitualPortal.com:

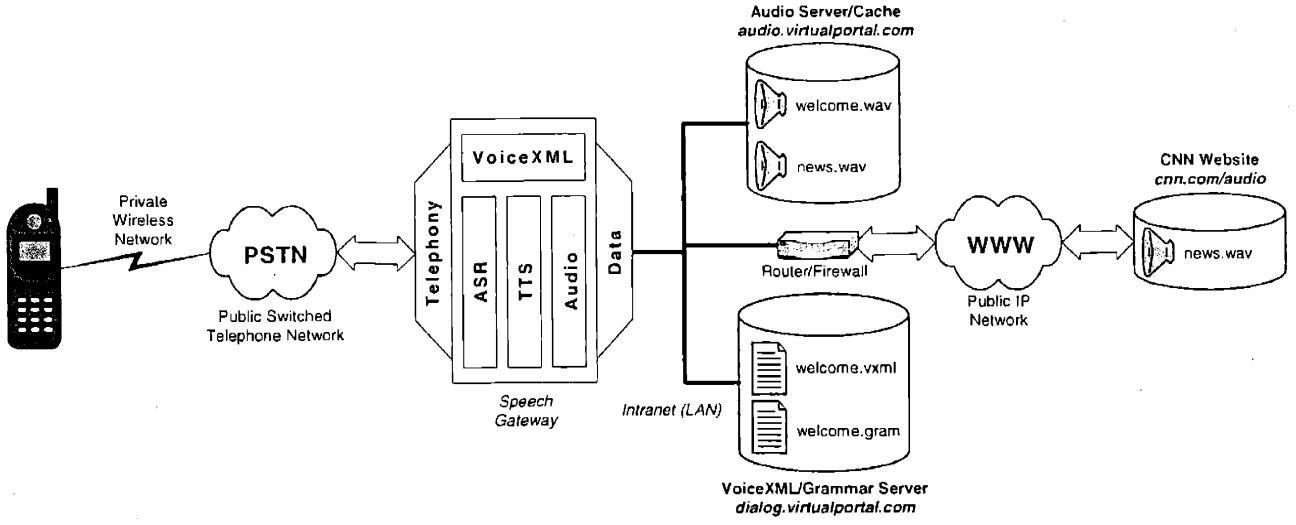

Figure 32. SIT Architecture for VirtualPortal.com Example

8.1.1. Internal Content

[126] The word *portal* is used in this context to describe a business model: consumer access to various types of data through a single aggregator of external sources. It has also been used (incorrectly, in the author's opinion) to variously describe the VoiceXML interpreter, the entire speech gateway, the business of outsourcing speech gateway capacity, and even just the speech-accessible element of any business (not necessarily content aggregators).

Internal Content. To trace the flow of data through the architecture, suppose that upon receiving a call, VirtualPortal issues the following message (previously recorded and stored in http://audio.vitrualportal.com/welcome.wav): "Welcome to VirtualPortal. Please select news, sports, or weather." One of VirtualPortal's many audio offerings is the CNN Headline News, previously recorded and published by CNN itself (stored in http://cnn.com/audio/news.wav.)

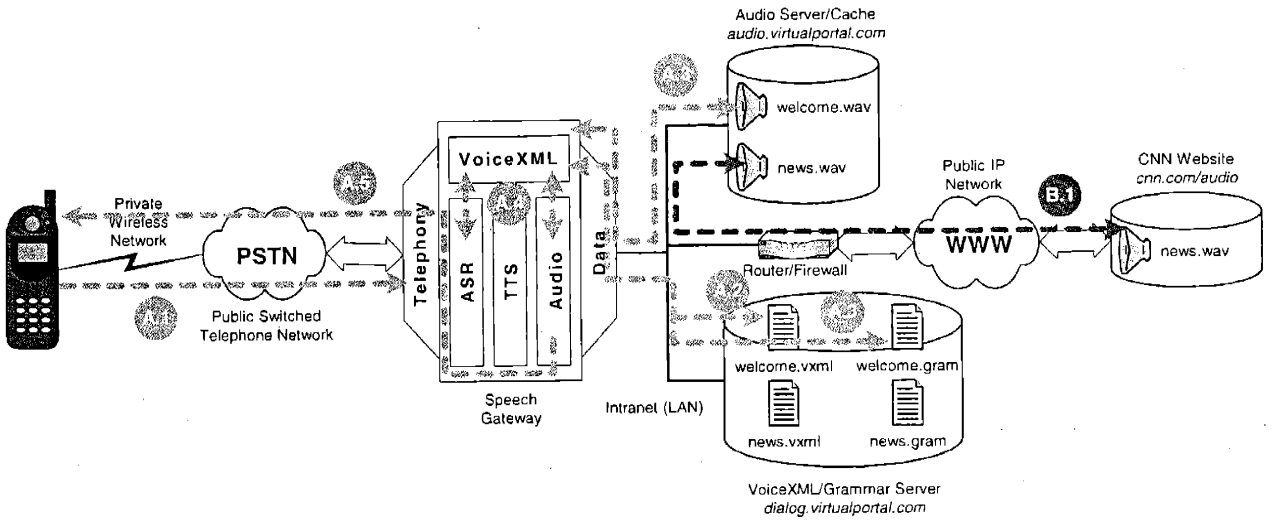

Figure 33. Dataflow for Internal Content

There are actually two processes involved in servicing the caller: the real-time process of issuing prompts and interpreting responses (Process A in the diagram), and the asynchronous, offline process of ensuring that external static content is locally cached to avoid real-time playback over the Internet (Process B.) Following the flow through the diagram:

- Step A.1: The caller dials the number to access the speech gateway through the PSTN.
- Step A.2: The gateway assigns an instance or thread of the VoiceXML interpreter to this unique caller, and the default starting page (http://dialog.virtualportal.com/welcome.vxml) is requested from the VoiceXML/Grammar Server and loaded into the browser.
- Step A.3: Like most HTML web pages, the VoiceXML document is composed of markup text and links to other resources, such as the grammar file (http://dialog.virtualportal.com/welcome.gram) appropriate for this dialog and the audio file (http://audio.virtualportal.com/welcome.wav) for the prompt. A series of http requests from the VoiceXML interpreter fetches these resources from across the intranet.

- Step A.4: The grammar file is then loaded into the ASR engine in anticipation of the caller's speech input, and the audio file is loaded into the audio engine for formatting and playback functionality.
- Step A.5: At the earliest opportunity, the audio file is modulated onto the PSTN by the telephony interface and ultimately reaches the caller's telephone.
- Step B.1: At some arbitrary point before the call arrives at the speech gateway, the audio cache stores a copy (http://audio.virtualportal.com/news.wav) of the remote file located on CNN's servers (http://cnn.com/audio/news.wav.) The frequency interval cannot be so large that callers are not likely to get the latest version of the file (an important issue given that phonecasting's utility lies in large part on timeliness)-but not so short that the connection to the Internet is inefficiently utilized and the audio server's performance decreases under an unnecessary load.

For readers familiar with traditional web technologies, these processes should appear similar to those involved in conveying (http) and presenting (html) web documents. The major difference is that interpretation is handled by the service provider hardware, but this a matter of location and not functionality. Before continuing the example, a few items of clarification:

- The diagram depicts the caller using a mobile phone, but the flow and processing involved in service the call do not depend on the end user device.
- The caching process is entirely optional: the CNN audio file can be readily requested via http from across the public Internet and loaded into the interpreter. The audio cache provides only a performance boost, but it is an important one given the unsuitability of the Internet to reliably convey audio in real-time. Note that the audio cache is typically part of the audio server itself (or at least an adjunct processor to the audio server), and not a component of the speech gateway.
- Related to the audio caching issues is streaming: the example assumes static audio files for simplicity, but the same configuration could be employed to support streaming audio, across the LAN and even the Internet.
- The number and arrangement of servers and devices is the purview of the individual service providers. For example, the VoiceXML and audio severs can be combined into a single server-the LAN and server network is designed to eliminate performance bottlenecks and resilience to element failure, so a single-server configuration is often avoided.
- The example assumes that all the prompts are professionally recorded, but they could also be the result of TTS synthesis from a text prompt: the only modification to the diagram would be to direct the text to the TTS engine.

- The example assumes that the VoiceXML and grammar documents are static and always available on the dialog server, but the SIT application programmer has the same tools as her HTML cousins to dynamically generate code in real-time. For example, a Java servlet on another server can request a data-populated bean from a database, and then forward that bean through a JSP markup template to generate the VoiceXML on the fly. The diagram can be expanded to any number of concurrent VoiceXML generation strategies without changing any of the existing flows.
- The central orchestrating role of the interpreter is a significant departure from traditional IVR architectures. A traditional, standalone configuration can have custom pathways and datatypes, avoiding the need for a strong integrative resource. In addition, the traditional IVR's business logic almost always resides on the IVR itself and is inherently static, so the complexity of managing remote logic is noticeably absent. In contrast, given the inherent flexibility in network configuration, remote business logic generation, data location, and protocol/file formats, the VoiceXML browser is the heart and soul of not just the speech gateway itself, but of all the elements that touch the Voice Web.

8.1.2. Cached External Content

To continue the VirtualPortal.com example and demonstrate speech recognition *in situ*, suppose that the caller responds to the menu prompt with "News," setting the system in motion to retrieve and playback the CNN Headline News audio segment.

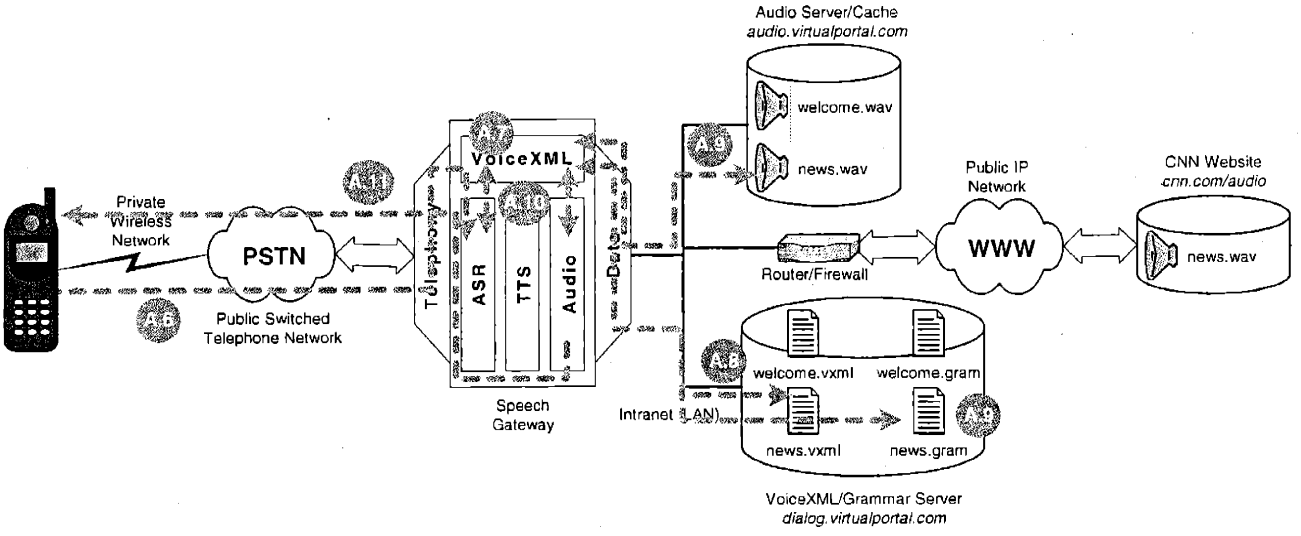

Figure 34. Dataflow for Cached Content

- Step A.6: The caller speaks the word 'news,' which is conveyed over the PSTN to the telephony interface. The signal is determined to be a valid speech event, so the VoiceXML browser forwards it to the ASR engine for segmentation and matching, using pre-loaded grammar file to increase performance and efficiency.
- Step A.7: Once the ASR engine has made a determination, the text result is returned to the VoiceXML interpreter. Upon learning that the caller had spoken the word 'news,' the existing VoiceXML document indicates that the next link to follow is found at http://dialog.virtualportal.com/news.vxml. This is completely analogous to the logic involved in processing a mouseclick on a HTML hyperlink: the recovered text indicates which link to follow.
- Steps A.8 and A.9: Just as with the welcome.vxml request and transfer, the news.vxml page on the dialog server is loaded onto the VoiceXML interpreter, triggering further http requests for the CNN audio segment (http://audio.virtualportal.com/news.wav) and the next active grammar (http://dialog.virtualportal.com/news.gram.)
- Steps A.10 and A.11. As before, the grammar and audio files above are loaded into the ASR and audio playback engines, respectively, and the audio subsystem drives the playback through the telephony interface and ultimately to the cellphone's tiny speaker.

In Step A.7, it should be noted that just as with HTML, the VoiceXML page can be written with the service logic embedded inside, or the page can be instructed to simply return the ASR result to the dialog (or other) server for service logic processing. This is particularly important when the "next" action is contingent upon information in addition to the ASR result alone. Developers who deploy complex sites rely on state based systems to implement the service logic, so simple result-reporting in VoiceXML is yet another example of the divorce between the presentation and application layers.

Note that neither of the two scenarios above depicts a Voice Web business model. The content was either developed in-house or licensed from an external agency (e.g. CNN), but this is hardly the universal web of VoiceXML content that the Voice Web envisions. This is simply an extension of the "MyYahoo!" or "AOL" concept to the speech-interactive telephony market: a single business entity creates and/or aggregates content for presentation to the user. In this model, the user does not really interact with the external providers; **in other words, the caller is only accessing licensed content, not their voice site.** This market concept may in fact become the dominant model in the years to come, but it is certainly not the vision of the Voice Web: a single call navigating both within and between voice sites.

8.1.3. External Markup and Content

To illustrate the flow for a Voice Web market, suppose that the caller interrupts (barge-in event) the playback of the CNN Headline News with the utterance "weather." This barge-in event is detected and the word is recognized, and a request is made to load the VoiceXML document for the Accuweather meteorological forecast application. Unlike the CNN news example, the Accuweather assumes responsibility for developing and serving the weather application from across the Internet, so the model shifts from a license-and-republish view to a direct-link view: all of the service logic and content resides with Accuweather.

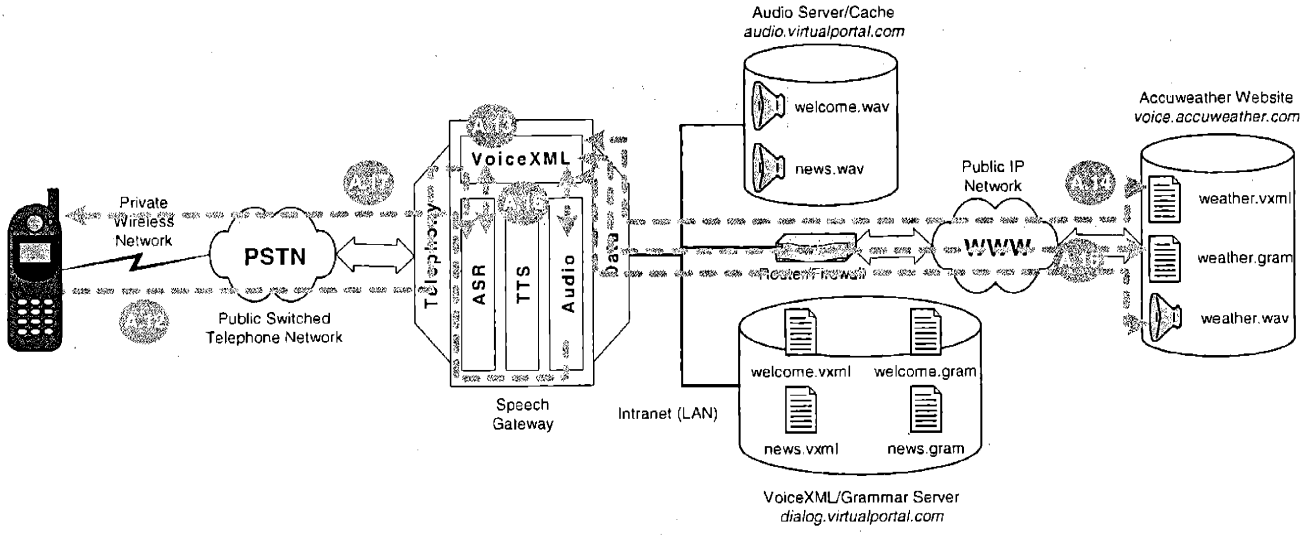

Figure 35. Dataflow for External Content

- Step A.12: The caller's "interruption" is conveyed through the PSTN to the telephony card, triggering a speech event signal to the interpreter.
- Step A.13: The browser routes the signal to the ASR engine for comparison against the pre-loaded grammar. Since the pre-loaded grammar is indeed designed to intercept a legitimate request for other content on the audio portal, the ASR engine returns the text 'weather' to the interpreter.
- Steps A.14 and A.15: The 'weather' results extracted from the speech input triggers an http request to a remote VoiceXML document on Accuweather's servers: http://voice.accuweather.com/weather.vxml. The document is returned across the Internet in real-time and loaded onto the interpreter, triggering the requests for the audio prompt

(http://voice.accuweather.com/weather.wav) and the new grammar to support the service logic (http://voice.accuweather.com/weather.gram.)

- Steps A.16 and A.17: The interpreter loads the ASR and playback engines with the new files, and the prompt is played out through the audio subsystem and ultimately reaches the caller's ears.

While the difference in locations makes no functional difference to the system's operations, the performance may be drastically impacted. Not only are the audio files transmitted over the Internet in real-time, but also the presentation logic and the grammar-the CNN example involved only infrequent, offline requests to a single file across the Internet, whereas the Accuweather example involved two round trips across the Internet in real-time.[127] Not illustrated in the picture but similarly impacted by latency, remote streaming audio files would also suffer degradation during the trip across the public network.[128]

The simplicity of the diagram obscures perhaps the most crucial aspect of incorporating external VoiceXML applications: standards. Despite industry movement toward standardization, VoicePortal's speech gateway network can be flexibly implemented: energy-based vs. recognition-based barge-in, pause feature for long-play audio, language format for TTS synthesis directives, grammar format, support for RealAudio vs. ASF, streaming-capable cache vs. standard cache, etc. This engineering flexibility in the absence of minimum standards potentially destroys the network externality effort for third-party developers: How can a content company design its VoiceXML application if there are no baseline assumptions for feature functionality? Should the company develop a custom site for each speech gateway network over which it wants to distribute? Wouldn't it just throw up its hands and prefer to collect licensing fees? When the presentation logic (i.e. VoiceXML documents and grammars) is developed in-house using external content, the only content challenge is to cache it in the proper format. But once the floodgates are opened to third-party developers, these external programmers need an intimate knowledge of the features and limitations of that particular speech gateway installation. The reality is that the adoption of the VoiceXML standard may not be enough to ensure the formation of the Voice Web.

8.2. Summary

[127] There are actually three round-trips, but the transfer of the grammar and audio files occur in parallel.
[128] In fact, live streaming audio is a problem even if the VoiceXML and grammar files are proximal to the speech gateway, since the content must be streamed in real-time over the Internet. Without a streaming-aware caching server, the problem expands to include all streams, both live and pre-recorded.

Recognizing that the pursuit of the knowledge of the speech gateway and its component technologies may cause the reader to lose sight of the bigger picture, the goal of this chapter is to integrate and reinforce these isolated ideas within the larger context of a functional architecture. The proverbial devil is in the details, and an appreciation of these concepts will serve the reader well, especially for the sections that explore the role of technology as both enabler and encumbrance to this industry. However, the following systemic observations distill the preceding technology discussion to a few key points:

- **Visual Web Inheritance**. Given the prevalence of the Internet and the HTML Web as the preferred delivery vehicles for both services and products, it is no accident that the next-generation speech-interactive telephony network borrows many familiar elements: servers, databases, packet networks, even code generation and management tools. It should also be appreciated that not all speech applications are well implemented using the client-server model, just as not all visual applications are well suited to the Web (imagine Microsoft Word for the Web?) Users of visual web pages are forgiving of latency, but the telephone has created an atmosphere of intolerance for dead air during a phonecasting experience, so the phonecasting architects are driven to eliminate delays wherever and whenever possible.
- **Speech Gateway**. Perhaps the only truly novel and distinguishing element of the SIT architecture is the *speech gateway*, which is responsible for understanding speech input, driving audio output, and otherwise determining the nature of the interaction with the caller at every dialog. The speech gateway is also unique in that it assumes the role of interpretation (a.k.a. execution) within the service provider network, a task normally assigned to end user devices. Therefore, the speech application service provider experiences an operational and financial burden unknown to its HTML-based contemporaries.
- **Input "Problem."** Despite the all the technology and application elements that can be borrowed from the HTML industry, the fundamentally unconstrained nature of the speech input complicates the development process. Grammars can be applied to assist the ASR engine in making an appropriate match, but they neither guarantee 100% accuracy nor restrain the user from saying whatever she wants to. This input model is vitally different from that of a drop-menu or button on an HTML or WML site, in which the input is recognized with 100% accuracy every time and the user is totally restrained from making any choices outside of the ones proffered. Advances in ASR performance and accuracy are worthwhile in that they positively affect the user experience, but the unconstrained nature of the input cannot be overcome by such improvements.

- **PSTN Inheritance**. While the use of existing wireline and wireless telephones eliminates the purchase of an expensive device to experience phonecasting applications, the PSTN network imposes restrictions that affect both the cost of the architecture and the quality of the consumer experience. While the circuit-switched orientation of the telephone network provides latency-free transmission of audio, it comes as a high component and network cost: each circuit, whether physical or virtual cannot be multiplexed among several concurrent calls. The end result is that the resources of the telephony interface are not efficiently shared among the consumer base, putting an additional burden on the revenue model to recover the interface and telephony costs on every call.
- **Audio Fidelity.** Another consequence of utilizing the PSTN infrastricture is the 64-kbps limit on audio transmission, an engineering assumption carried over from the days when the only traffic was human conversations. Despite the obvious degradation in audio quality and attendant reduction in consumer utility, the abundance of branded audio content already on the Internet provides fertile ground on which to plant the seeds of the Voice Web; however, this content is usable only if speech gateway vendors finally make a commitment to undertaking the task of integrating streaming audio technology into their platforms. Furthermore, the amount of audio on the web is only a fraction of the text-based content on the web, but the general consensus is that TTS technologies do not properly simulate natural human speech: there is a tension between the availability and low cost of textual content, and the potentially consumer-offensive TTS output for these text sources.
- **Standards**. If open standards are essential to the formation of a vibrant third-party network of services and products, then the introduction and early adoption of VoiceXML as a presentation language standard is definitely a step in the right directions. However, it is easily demonstrated that the gains made by VoiceXML can be offset by the absence of other relevant technology standards: grammars, TTS annotation, ASR features, etc. The introduction of a language alone may be sufficient to generate general interest in developing stand-alone applications, but the creation of a Voice Web requires a much more intimate exchange of data and logic across organizational boundaries-hence the need for standards.

Even apart from the technology, there are a variety of economic and regulatory factors that affect the nature and speech of development for the speech-interactive telephony market. Whether the reader's goal is to shape market by manipulating these factors, or predict the outcome based on the current parameters, the rest of paper is dedicated to exploring the interplay between technology, economics, and regulation in the development of a sustainable Voice Web.

PART II. STRATEGIC ANALYSIS

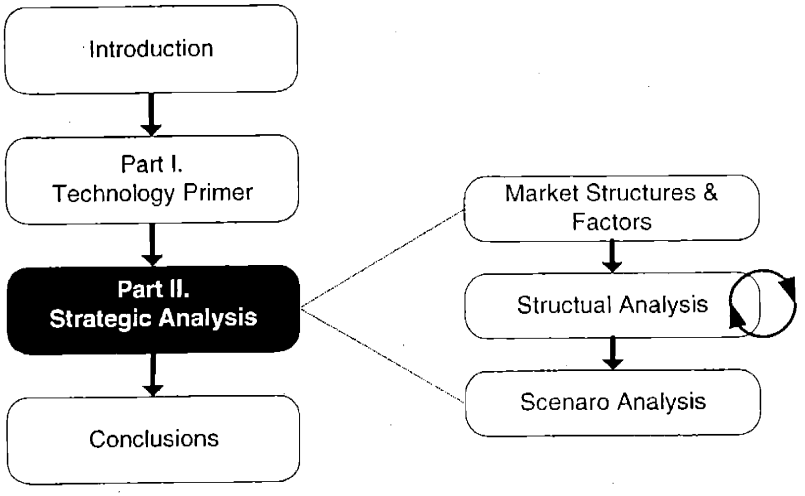

Figure 36. Overview of Paper (Currently on Strategic Analysis)

"A strategic position is a path, not a fixed location."

- Michael Porter, *Competitive Strategy*

In the hopes that the preceding Technology Primer has provided a secure understanding of the SIT architecture, the reader can profitably engage the Strategic Analysis in Part II, which is developed as follows:

- **Market Structures & Factors.** Similar to the Reference Model for Program Interaction in Part I, the Strategic Analysis will begin by constructing a Reference Model for Market Structure and enumerating the market factors that will be applied to it.

- **Structural Analysis.** Given the Market Model and Factors, each segment will be considered in isolation as to how the presence or absence of certain factors affect the formation and development of the Voice Web (either positively or negatively). Because of the multi-tiered market structure of the voice industry, this is necessarily an iterative process.

- **Scenario Analysis.** Considering the enabling and disabling factors discovered in the Structural Analysis, various market configurations (e.g. natural points of integration and disaggregation) are explored for a variety of time horizons.

Chapter 9. Market Structures & Factors

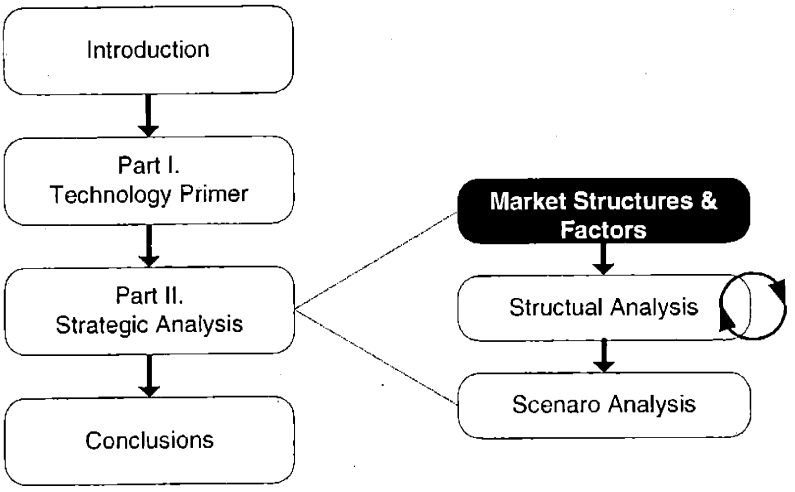

Figure 37. Overview of Paper (Currently on Market Structures & Factors)

In many respects, the voice market is not dissimilar to the traditional broadcasting arena in that there are a variety of public and private organizations jockeying for position in the value chain. The previous chapter outlined the major technological components required to support a SIT application, but nothing was implied regarding how those functionalities should be apportioned. Especially in an embryonic market such as this, the vertical and horizontal levels of integration and disintegration are far from obvious, as evidenced by the multiplicity of business models for the firms that have entered (and in some cases, already exited) this space. Some firms already unwittingly participate in this chain by creating audio content and posting it on the Internet for another purpose, and others deliberately attempt to achieve complete vertical integration from content creation to distribution. The particular configuration of the market is not simply a matter of shifting money flows: the market structure will determine the nature of the product that will ultimately be delivered to consumers.

AOL vs. WWW. To demonstrate the far-reaching affects of market configuration, consider the evolution of the online content delivery industry into two major segments: America On Line (AOL) and the World Wide Web (WWW.) AOL best exemplifies the virtues of the *walled garden*: vertically integrated into both access and programming, completely private network (e.g. AOL channels are not accessible from outside the AOL network), keywords and channels are expensive commodities, 100% observation of the user clickstream, proprietary document standards, etc. In recent years AOL has allowed users to browse the Internet via an internal HTML browser, however, many sites will not view or function properly

because AOL employs an older, customized version of Internet Explorer[129]-many sites with an AOL keyword presence opt to generate a custom HTML site for AOL viewers. Given the closed nature of the network, it is not surprising that consumers and content/service providers were unwilling to sustain the three major, vertical service providers in this area: AOL acquired CompuServe, and Prodigy evolved into a standard ISP. It is this 'walled garden,' however, which encompasses many of those features that attract consumers to AOL: easy-to-use keywords instead of cumbersome URLs (many ads proudly list the AOL keyword alongside the URL), consistent and well-designed user interface throughout the entire AOL space, highly-targeted content and advertising based on consumer preferences, integrated wallet functionality throughout all of the AOL e-commerce shops, extensive customer support for all facets of the online experience, etc.

Contrast this market configuration with that of the traditional HTML web: market segmented into ISPs and WWW peers, privately-owned but publicly-accessible open-standards network (all sites accessible from anywhere on the public Internet), domain names are easily obtained for less than $30 per year (subject to availability and trademark restrictions), no single peer site can observe the entire clickstream, public markup language standards, etc. The Internet market is not only vertically-disintegrated, but there is strong competition in both the access and programming arenas. The ISP community has little or no financial relationship with the site providers, so users are not actively discouraged from exploring the entire WWW. However, all this flexibility comes at a cost: URLs lack the intuition of keywords, each site can sport any randomly-selected user interface it chooses (it can even forgo HTML and just use Macromedia Shockwave to implement the entire site.) Since there is no universal profile or wallet information across sites, services like targeted advertising and e-wallets are extremely difficult to implement and sporadic in their coverage.

9.1. Four Market Factors

As the case above clearly demonstrates, market structures affect not only monetary flows, but they also determine the very nature of the consumer service. These lines of organizational aggregation and disintegration depend on a variety of *market factors*, exogenous (i.e. inter-organizational) and endogenous (i.e. intra-organizational) stimuli that shape the strategic positioning of each firm. These market factors come in many forms: *economic*, *technological*, *human*, even *regulatory*. Most firms mistakenly consider

[129] See http://webmaster.info.aol.com/

strategic thinking as an exercise in preserving their existing business models through cost minimization and revenue maximization, ignoring the more subtle factors at their own risk.

For example, since the WML and HTML webs so closely resemble each other in terms of architecture (and even service, to some extent), one might assume that the WAP industry would assume a configuration similar to that of the established HTML Web: peer sites operating over an agnostic network. Nothing from an *economic* or even *technological* analysis would reveal evidence to the contrary, but other factors are indeed at work. From a *regulatory* perspective, wireless communications firms do not operate under the same rules as their wireline counterparts: they have no obligation to carry data traffic or even allow general access to the Internet. Unlike the multiplicity of competitive ISP options (irrespective of the provider of telephone service) available from the consumer's home computer, the cellphone will only work on a particular carrier's network, and only that carrier can provide data service. The wireless carrier could simply choose to restrict the user to the WML sites listed on the main menu of the browser; however, the carrier doesn't even need to exercise this right thanks to the presence of a *human factor*, specifically, the user interface. Since it is so cumbersome to enter a full URL address using only the keypad on the phone, the vast majority of users simply rely on the menu choices provided on the carrier's default page. This fact is not lost on the carriers that charge large sums for top spots on the default menu, reducing the effective WAP field to those with the wherewithal to pay off the cellular service providers.[130] So even though the WAP phone has the technology architecture to access any WML site on the Internet, the presence of these market factors allow the carrier to create a closed, "AOL-like" experience-certainly not the WML Web envisioned by the media and many others.

While it certainly isn't the case that all factors lead to such market-defining events, the firm's strategists would be remiss if they failed to identify them.

- **Economic**. If a carrier's cellular network is already being strained by the load during commute times, does it even make sense to deploy an audio product that encourages long-hold times?
- **Technological**. How can an audio portal allow users to access external VoiceXML sites without solving the public Internet latency issue?
- **Human**. Who is going to enforce a portal's global keywords on external VoiceXML sites, especially when there is no consensus of what those keywords are?

[130] InformationWeek.com, News, Carrier Strategy Limits WAP Phone Users' Online Choices, 30 October 2000. See http://www.informationweek.com/810/wireap2.htm.

- **Regulatory**. Can a portal using real-time HTML-to-VoiceXML transcoding publicize that it has CNN and ESPN on its portal, or does it need to license the content? In other words, is the portal really just an agnostic ISP, or a destination site that is stealing content?

9.2. Reference Model for Market Structure

In order to have a meaningful discussion about how these four factors shape the structure of the industry, the elements subject to this process of integration or disintegration need to be defined. More specifically, each factor can be considered as a *vector,* expressing both scale and direction. For example, the economic factor of incoming minute costs certainly affects the voice market, but in a very specific arena such that a firm that hosts an audio portal on behalf of a large carrier is largely unconcerned with the problem (the cost is simply passed on to the client), whereas a portal that markets directly to consumers is obsessed with reducing that recurring cost. These factors impact the value chain differently in both magnitude and point of contact.

Building on the now familiar Reference Model for Application Interaction in Figure 5 (page 28), the *Reference Model for Market Structure* enumerates the elements of the value chain from application provider to end user.

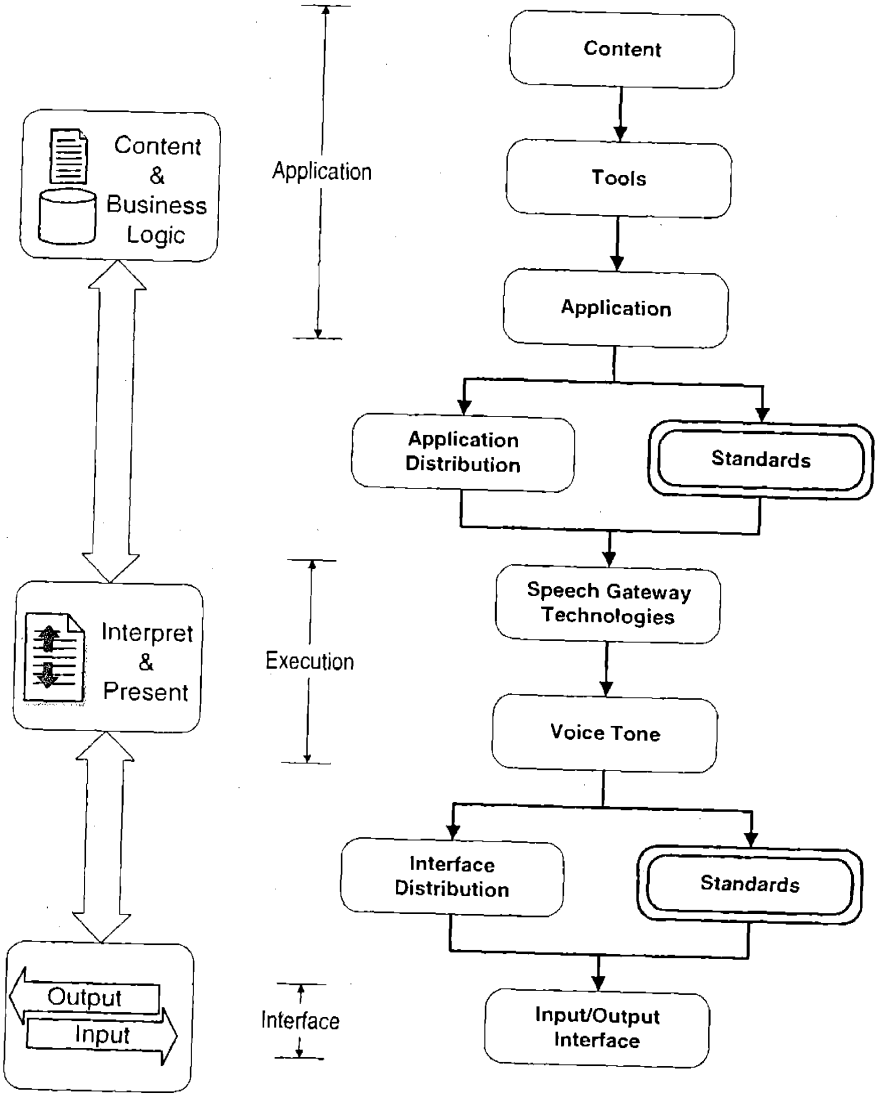

Figure 38. Reference Model for Market Structure

While each of this elements could arguably be further segmented, the essential market segments are represented and provide sufficient granularity for discussing the evolution of the voice industry, particularly with regards to the Voice Web model. These market structures flow quite naturally from their technology counterparts:

- **Content**. Sourced in either text or audio form, content can be a simple as a short prompt, or as infinitely lengthy as a continuous, live radio feed.
- **Tools**. Programmers always have a wide selection of tools to implement an application: hand-coding in a text editor, Java/dynamic-XML, database methods, etc.

- **Application**. Depending on the content, the application (or business logic) may simply act as a placeholder for an audio file, or implement a highly-interactive site with dynamic content.
- **Application Distribution**. The interaction between the application and the voice platform may span only a few feet or bridge large geographic distances, but whatever the proximity, the network must be able to support latency and throughput demands of the voice application.
- **[Application-Speech Gateway] Standards**. In order for the application to engage the speech gateway, there needs to be considerable agreement in terms of communication protocols (e.g. HTTP/TCP/IP) and document formats (e.g. VoiceXML, grammars, synthesis.) While the technology chapter demonstrated that an interpretable language would be more appropriate for consumer phonecasting services, this model can represent proprietary and even compiled languages just as easily.
- **Speech Gateway Technologies**. The Technology Chapter enumerated the various functional modules of a speech gateway: ASR, TTS, Audio Player, VoiceXML interpreter, telephony and data interfaces. One must also not overlook the considerable engineering task of integrating these various subsystems into a functional whole, so the speech gateway itself is represented by this element in the value chain.
- **Voice Tone**. An operational speech-interactive telephony network consists of speech gateways and elements commonly found in a HTML serving facility: intranets, databases, document servers, routers, redundant data and telephony switching elements, uninterruptible power supplies, consoles, telco racks, collocation space, etc. An organization that assumes these responsibilities essentially provides the core functionality for speech applications: *voice tone*. (as opposed to traditional *dial tone*.) The Voice Tone Provider (a.k.a. Voice Service Provider) is responsible for setting up the shop and running the system 24/7/365.[131]
- **Interface Distribution**. Depending on the type of end user device, access to and from the Voice Tone facility may be a stream of packets or a traditional circuit-switched PSTN signal. Even for a given output type, a variety of owner/operator configurations and hybrids are possible: public[132] PSTN network coupled with a private wireless network.

---

[131] The author prefers the term *Voice Tone Provider* (VTP) to *Voice Service Provider*(VSP) because the former implies a base functionality in support of external applications. Given the possible confusion between 'services' and 'applications,' the latter term may be misinterpreted to mean that the VSP itself is responsible for implementing the application. However, VTP and VSP will be used interchangeably throughout this paper.

[132] To reiterate, the author's use of the word *public* implies compulsory common carriage, not communal or governmental ownership.

- **[Voice Tone-I/O Interface] Standards**. Since the Interface Distribution is simply sending and receiving audio signals, the task for which telephony networks have been designed, the only requirement is that the PSTN-formats are respected and supported throughout transit.
- **Input/Output Interface**. While this paper focuses on the speaker and microphone of the traditional wireline or wireless telephone as the primary I/O interface, the need to overcome limitations (e.g. audio fidelity) invites the creation of alternatives.

Since these two elements do not arise from the dataflow suggested by the value chain above, they are omitted from the diagram above; however, their absence should not imply that they are no less important strands of the voice market:

- **Marketing**. Unlike an enterprise application, a phonecaster must take its message to the people, so market research and branding are essential elements of deploying a compelling service. For example, a startup may choose to take their message directly to the consumer on its own, or a carrier might incrementally introduce the product to its existing customers.
- **End User**. The consumer is the recipient of the marketing and the ultimate arbiter of whether phonecasting has a future, so the discernment of her perceptions and needs are a priority. Given that consumers are already inundated with a variety of media sources, the introduction of yet another weather news/sports/stocks service[133] may entirely miss the market (?).

The two standards 'boxes' are highlighted differently than the other because they do not typically participate in the real-time flow of data, but they do represent a concerted industry effort to support broader distribution through alignment with *de facto* and *de jure* standards. Given the recent introduction of VoiceXML as a potential disintegrating force within the voice market, particular attention will be paid to their vector factors on the market overall. The reader should also note the explicit inclusion of the connecting networks between each of the 3 Application Interaction elements: the real-time performance demands of a voice application warrant individual examination of the capacity of existing networks and the availability of alternatives.

These elements do not imply any particular physical boundaries, nor do they necessarily require that all firms within the market assume the same configuration. For example, AOL Time Warner not only

[133] In the tradition of *WAPapthy* (WAP apathy), the author humbly contributes the following to the lexicon: Yet Another Weather, News, Stocks, Sports portal = ***YAWNSS.***

produces content (e.g. CNN News Group, Home Box Office), but it is also the owner of the second largest cable distribution network in the US (Time Warner Cable); in contrast, the A&E Network focuses exclusively on producing and aggregating content, such as the A&E and History channels. What happens when A&E approaches Time Warner for a slot on its network is a function of many factors: does Time Warner offer a competing product, what is the consumer demand for A&E content, should A&E pay for channel space or should Time Warner pay to have such a popular channel on its network? The outcome when different organizations meet is determined by the confluence of salient market factors, all of which have scale and direction.

9.3. Applying the Model to the Visual Content Market

Though the Market Model has been tailored to the represent the element of the voice market, it is equally capable of representing the avenues of segmentation and integration within the more familiar HTML Web market:

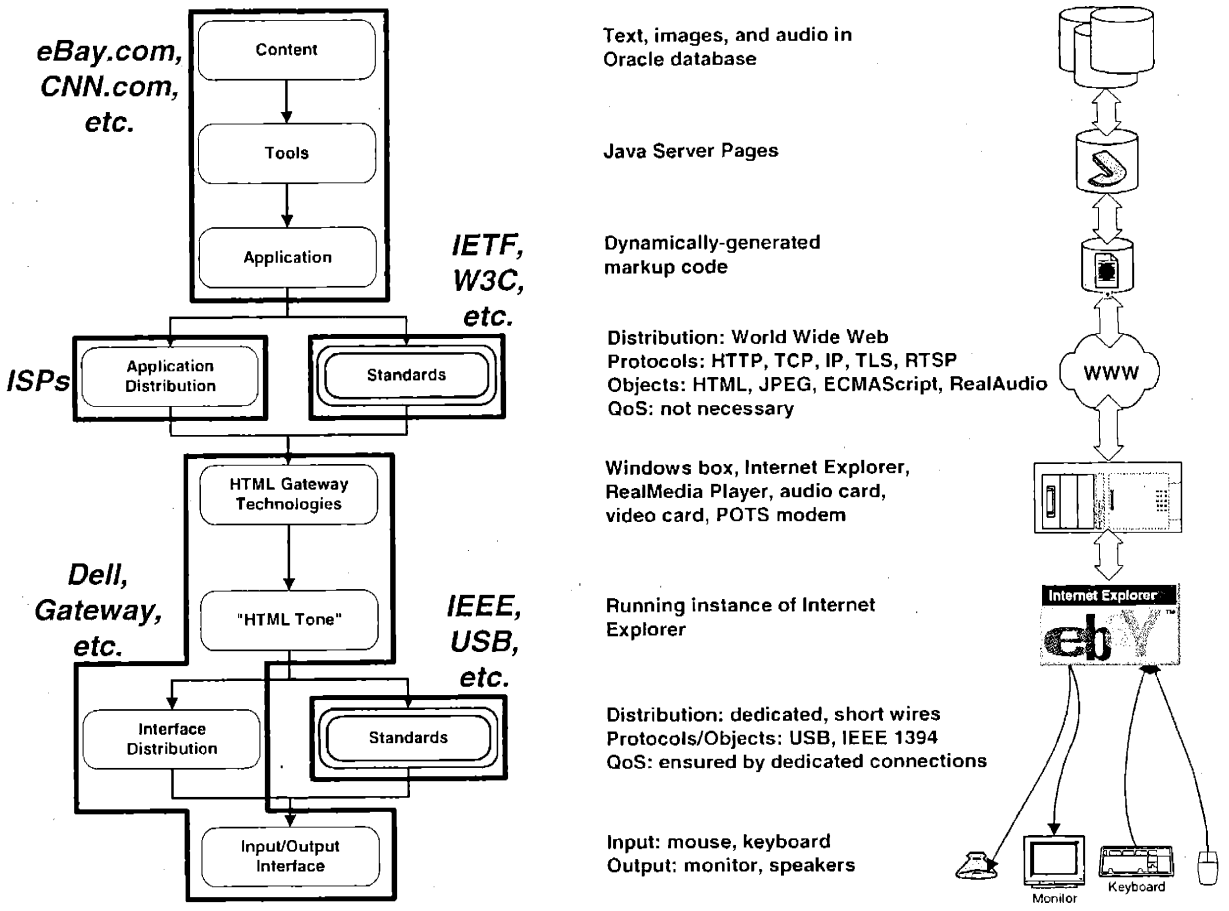

Figure 39. Market Structure for HTML Web

- **Web Sites**. Most firms on the web take direct responsibility for delivering their HTML content, using either in-house or contracting resources. The HTML market has matured sufficiently such that there are a variety of technologies to manage the creation and publication process, particularly for dynamic and personalized content.
- **ISPs**. The familiar public Internet is the conduit between the Application Provider's content and the End User. HTML applications are designed such that the variable latency and throughput of the public Internet does not affect their usability or utility. Application firms and end consumers alike have many choices for Internet access within a competitive connectivity industry.
- **Web Site-Personal Computer Standards**. The communications protocols (e.g. HTTP, TCP, IP, etc.) are standardized by the Internet Society's (ISOC) Internet Engineering Steering Group, which considers proposals from the more familiar Internet Engineering task Force.[134] The ISOC's

[134] See http://www.isoc.org/standards/.

Internet Assigned Numbers Authority (IANA)[135] is responsible for overseeing the crucial elements of domain name registration and IP address assignment. The document protocols (e.g. HTML, XML) fall under the auspices of the World Wide Web Consortium.[136] Major image and moving picture formats have been codified by the International Organization for Standardization (ISO)[137], including the MP3 audio codec[138] and the JPEG image format.[139] Apart from these large standards bodies, there are a variety of smaller open or proprietary protocols and formats that are typically encountered on the web: RealNetworks G2 codec and Real Time Streaming Protocol[140], Adobe Acrobat, Macromedia Shockwave and Flash, European Computer Manufacturers Association's[141] ECMAScript, etc.

- **Personal Computers**. Unlike VoiceXML's limitation of remote interpretation, the computer system connects directly to the application distribution network and handles all aspects of HTML interpretation (i.e. execution) and the interface with the end user. Except for establishing a connection to the Internet, nearly all computer systems are capable of natively supporting HTML applications right out of the box. This does not imply that the PC manufacturers create all aspects of the system since they are obviously integrating technologies from a variety of sources; however, the consumer market is clearly oriented towards purchasing complete systems.
- **Personal Computer-Interface Standards**. Since the market is firmly focused on delivering complete systems, most computer owners are entirely unaware of the standards bodies involved in defining hardware and software specifications, e.g. Institute of Electronic and Electronics Engineers (IEEE)[142], Peripheral Component Interconnect Special Interest Group (PCI-SIG)[143], etc. It is worth mentioning that these standards are not held hostage by any single computer manufacturer.
- **Marketing**. Not shown in the picture are the various marketing efforts engaged by the market participants. Web sites aggressively market their sites to end consumers, and both consumers and app providers alike are targeted by the ISP industry. In addition, the end user is buffeted by the

[135] See http://www.iana.org/.
[136] See http://www.w3.org/.
[137] See http://www.iso.ch/.
[138] See http://www.cselt.it/mpeg/.
[139] See http://www.jpeg.org/.
[140] RTSP is currently a proposed standard at the IETF, published under RFC 2326. See ftp://ftp.isi.edu/in-notes/rfc2326.txt and http://www.rtsp.org.
[141] See http://www.ecma.ch.

[142] See http://www.ieee.org/.
[143] See http://www.pcisig.com/.

marketing efforts of the computer system manufacturers, since the user ultimately takes responsibility for ownership and operation of the HTML platform.

Despite the existence of certain close technologies, all protocol and format creators have made an effort to extend their availability to as many platforms as possible. Neither the content nor the personal computer industries wields these standards to the detriment of the consumer or the competitive market in general. Not only do the consumers and sites own and operate their own equipment, the ISP industry draws revenues towards the center of the network:

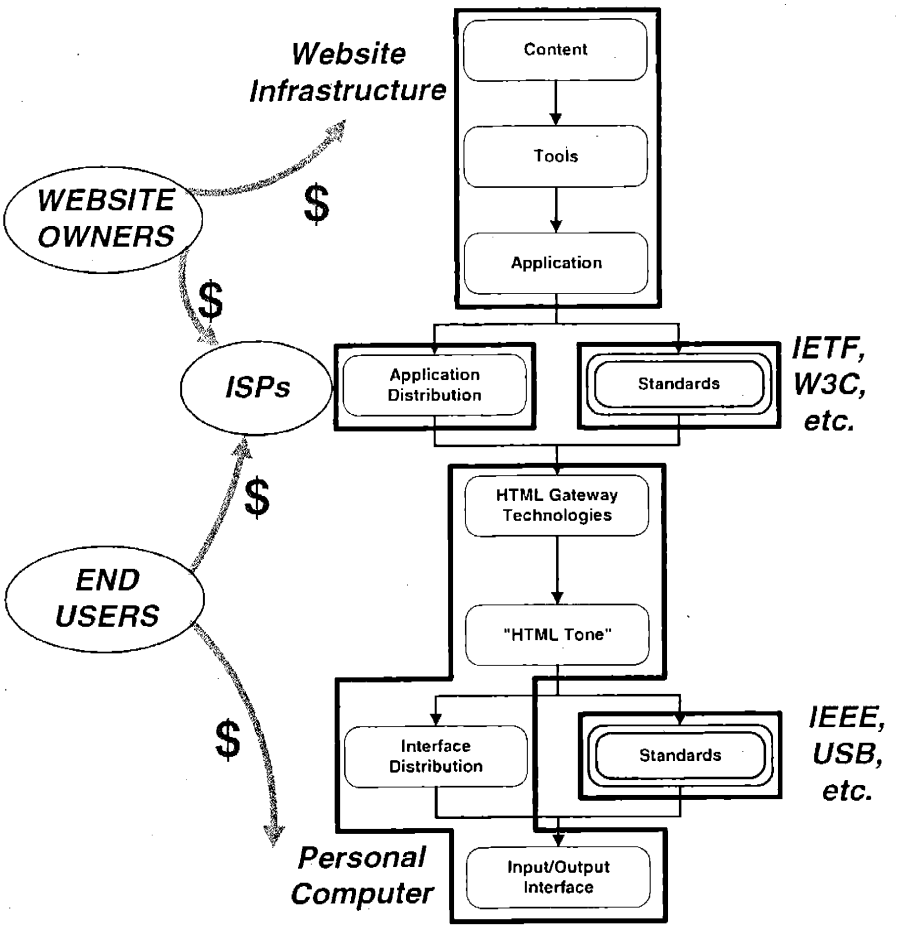

Figure 40. Monetary Flow for HTML Web

In contrast, consider the market model supported by AOL:

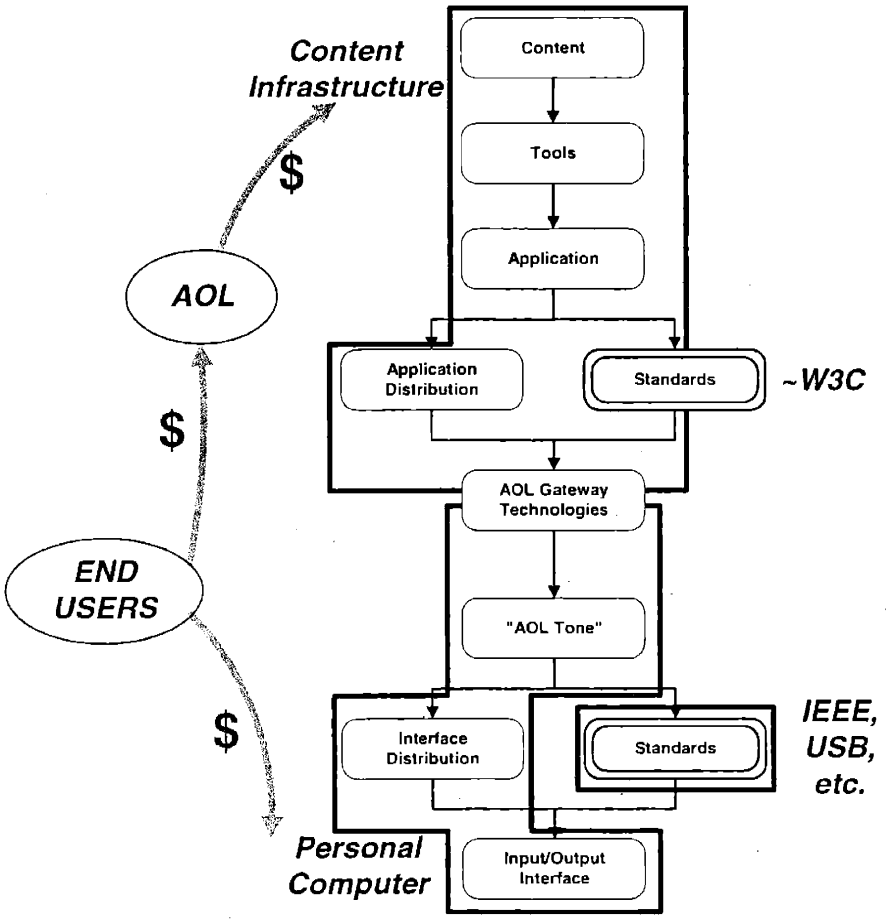

Figure 41. Monetary Flow for AOL Model

- **AOL: Applications, Application Distribution, Custom Gateway Technologies and Standards.** Instead of connecting through the Internet, AOL is reachable only through special dial-up connections. AOL itself develops most of the content and services encountered. AOL licenses keywords to point to other sites, but they must be custom formatted to deal with the technology restrictions of AOL's network: fidelity reduction of image files, support for only a subset of HTML tags, a proprietary program that must be installed on the target computer, etc. End users pay a premium above standard ISP rates to access AOL.
- **Personal Computers and Standards.** Apart from the installation of the proprietary interpreter, the computer system and related standards remain the same.
- **Marketing.** AOL obviously aggressively markets its services to consumers and carries sufficient market power to become part of the default configuration of many computer systems. The traditional computer system and content infrastructure marketing remains the same.

As indicated, the areas of aggregation and disintegration play a major role in the nature of the service and the sustainable number of players.

9.4. Applying the Model to the Consumer Voice Market

Unlike the HTML market, the levels of integration and disintegration within the consumer voice market have yet to congeal. Even if VoiceXML were to become the dominant dialog technology (and it has every indication of becoming so), this still would not imply a particular market structure. While the voice portals have dominated the media attention in the voice landscape, there has been little thoughtful discussion of what it even means to have a Voice Web.

Single-Point Model. For example, if the Voice Web's only criterion is that VoiceXML sites are available through special access numbers, then the world already has its Voice Web. In a sense, the telephone number becomes the canonical *domain name* for each destination on the Voice Web.

To Access	Visual Site	Voice Site	Applications
Charles Schwab	www.schwab.com	800-435-4000	Stocks
United Airlines	www.unitedairlines.com	800-824-6200	Reservations
MovieFone	www.moviefone.com	617-333-FILM	Movie tickets

From this perspective, VoiceXML serves only as a much-needed stimulus to encourage HTML and WML publishers to create voice sites, with the positive effect of growing the previously(?)/traditionally(?) parochial speech gateway industry. However, given the ineluctable movement within the telephony industry towards packet-based service networks, the circuit-switched telephone call seems an unlikely candidate for sustaining the Voice Web vision. This is redolent of the era of the Bulletin Board Systems: each BBS was typically dedicated to a single purpose or subject matter, and accessible only through special dialup numbers.

Portal Model. Given that consumers are more interested in receiving several content providers from a single provider than having to piece together their own disjoint network of products, it was only a matter of time before the BBS industry evolved into several large, unconnected content enclaves, such as AOL and CompuServe. While a special number was still required to reach each aggregator, that single call purportedly served all the subscribers' needs by allowing access to a variety of services. This model

proved enormously successful even as the HTML Web grew in prominence, but the proprietary nature of these networks ensured that monopolization by a single entity would be the ineluctable result.

The recent emergence of the voice portal closely tracks this evolution within the visual content industry: each portal is a closed system that offers a variety of consumer services. Again, a special phone number must be dialed to reach them, but navigation is speech-directed once the user is connected to the system-paralleling the shift from individual BBS numbers to a single number for AOL with keyword navigation.

To Access	Visual Site	Voice Site	Applications
BeVocal	www.bevocal.com	800-4-BVOCAL	Weather, News, Sports, etc.
TellMe	www.tellme.com	800-555-TELL	Weather, News, Sports, etc.

There are those that confuse this current state of the industry with that of a Voice Web, especially since the word *VoiceXML* is used so liberally in connection with these portals and the press they generate. However, this is no more a Voice Web than AOL is the World Wide Web: access, navigation, platform, and often content are vertically integrated within a single firm. Even as AOL began to open the floodgates to the larger Internet, it retained tight control over keyword assignment and which HTML tags it would choose to support through its custom browser. BeVocal and TellMe have been similarly encouraging content developers to deploy their applications on their speech networks, but they certainly retain their gatekeeper functions for their consumer portals and true code portability remains a fiction. In practice, there is very little difference between these voice portals and traditional hosting businesses, and the fact that multiple services can be reached through a single call is hardly an innovation of the voice portal (or even VoiceXML for that matter.)

True Voice Web Model. If the HTML Web provides the appropriate model, then VoiceXML's contribution will be the vertical disintegration of the market into applications, Voice Tone Providers, and platform vendors. However, this does not imply that all the necessary factors are aligned for delivering this particular market structure. As this chapter will demonstrate, the desire for a Voice Web must be tempered against the reality of the various factors: technological, economic, etc.

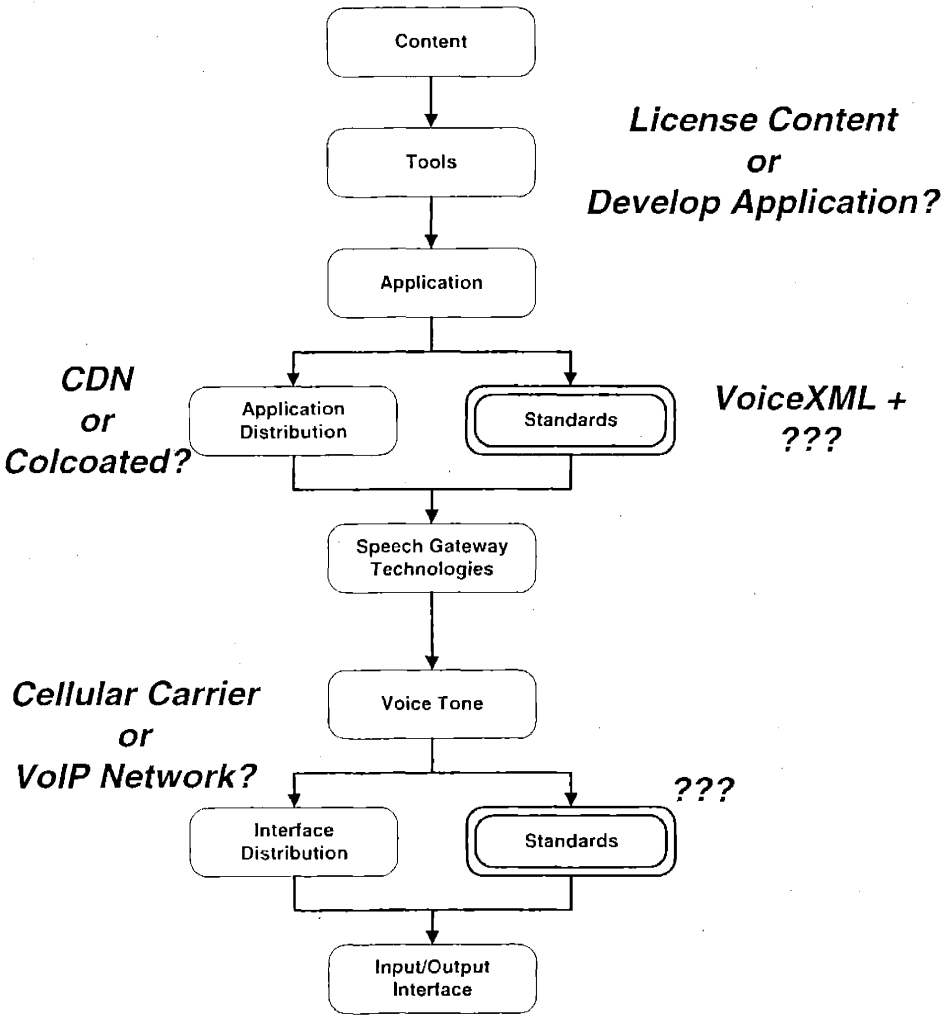

Figure 42. Market Structure for Voice Web

- **License or Develop**. Unlike the HTML and WML markets that provide a clear Web structure from the onset, the Voice Web's uncertainty creates real concerns for content publishers who may be more than willing to license their content to proprietary operators rather than to risk development dollars on an unproven product.
- **Content Delivery Network or Colocated**. Supposing that the content provider does indeed deploy its own application, must be it colocated with the speech gateway, or can it be served from the same location as its HTML and WML sites?
- **VoiceXML**. Despite the glowing press for this new specification, did its designers actually intend for it to be used within a Voice Web, or it is just a nice new language for standalone applications and portals? For example, what is the audio equivalent of a domain name and who will manage the registration of names?

- **Cellular Carrier or VoIP Network**. If Voice Web applications are geared to the mobile consumer, doesn't the carrier have an automatic lock on this market? Would callers use their precious cellular minutes to access these services?

While the path to single-point and portal solutions is much more clear, the viability of the Voice Web is far from clear. To address this and the more general issues of the voice industry, the Reference Market Model will serve as a guide in analyzing the various factors that restrict or expand the scope of any particular segment. To the fullest extent possible, each market segment will be analyzed in isolation and subjected to the 4-four factors analysis when appropriate. Each chapter will begin with matters of general relevance to the entire voice industry, and will end with a look at those barriers that arise only within the context of the Voice Web.

Chapter 10. Structural Analysis – Content & Applications

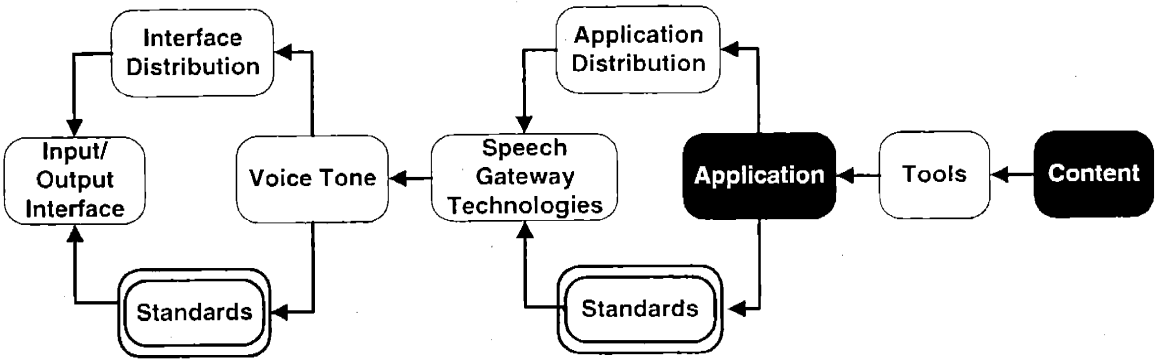

Figure 43. Market Structure (Currently on Content & Applications)

Despite the author's intention to examine each market segment in isolation, some linkages are too strong to decouple, which is indeed the case of content and applications. Prior to the introduction of VoiceXML, the common mode for voice application creation was to entirely outsource all facets of production from programming to hosting. Since these applications were often static or extended easily (e.g. enterprise employee locator), there was little need for the enterprise to engage the system directly or to even connect it to the corporate intranet. While this model may indeed continue to provide suitable service for many enterprise-oriented applications, voice applications will increasingly follow the pattern of HTML applications: business logic implementation and markup language generation is an integral part of the firm's resources. While web hosting and consulting services are certainly common practices, they usually serve an auxiliary role to the firm's core development team. The intent of VoiceXML was not only to stimulate voice services industry generally, but also to increasingly move the responsibility for code design and generation into the XML-savvy content provider camp.

Given that speech-related technologies have been around for many years, what are the reasons for the almost complete absence of consumer-oriented phonecasting services? VoiceXML certainly taps into the general familiarity of XML movement, but that alone cannot account for this industry-wide reticence. It is not the programming task itself that has proven the most insurmountable barrier, but the challenge of building and maintaining a voice tone facility. Even when the voice tone functionality is outsourced, the costs would prove prohibitive for services accustomed to deriving traditional advertising revenue from its various publishing efforts, e.g. whereas American Airlines would gladly undertake any measure to cut the costs of operating a live agent call center for reservations, the Weather Channel would be hard pressed to

justify offering ad-supported content over the telephone.[144] So the relevant factor is not that these services could not be created given the state of the *technology*, but that in the absence of a Voice Web that abstracts away the Voice Tone costs from the application providers, the *economic* considerations could not be surmounted.

Independent Content Provider Perspective. For organizations that already produce audio for video content for traditional broadcast or Internet distribution, content production is part and parcel of the normal business model. While the original source may be nothing more than a raw audio file on a tape, it is eventually configured for the appropriate protocol and format standards of the target network. The same reasoning applies to audio distribution: audio file format, transfer protocol, VoiceXML logic, grammar context, etc. The content provider then must make the decision to extend its organizational frontier to subsume these additional functions, or to allow an external agency to take control.

For example, when TellMe approached Zagat Survey with the offer of no-cost vocalization and application development in exchange for exclusive telephone distribution rights, Zagat could have determined that such work belongs in-house, such as with its HTML and WML sites currently. However, outsourcing mitigates a certain level of deployment risk, and the decision to bring production in-house can be reexamined at a later date. The other relevant topic issue is branding, e.g. what benefit or damage can occur by associating the Zagat brand with the startup TellMe?

These decisions are conditioned in large part by the perceived market configuration and the risk profile of the content firm. If Zagat perceives that the voice market technology is sufficiently immature such that it would have to develop custom apps for each SIT-architecture over which it desired distribution, then it would tend to allow audio portals to subsidize the cost of application development-Zagat knows that its brand brings cachet to the audio portal. In contrast, if MapQuest similarly infers that the voice market will not be a Voice Web but incompatible vertical architectures, but it wants to continue its tradition of tight control over branding and distributions, then it undertakes the task of developing two or more parallel systems.

In the absence of a true Voice Web, which produces both engineering and market stability, most content providers are likely to simply forgo voice application development or pray that their brand is strong enough to warrant considerable subsidization.

---

[144] In fact, the Weather Channel did indeed implement a Motorola VoxML version of a weather update service. They offered the service through both a toll-free number and a 900-number, both of which were loss leaders.

From an economic perspective, there is no doubt that most branded audio firms will gladly step back and simply license their content to willing audio portals. While most of these firms assume responsibility for supporting their own HTML (and sometimes WML) sites, they fully recognize that there is currently no Voice Web to accompany the VoiceXML standard. There is simply no justification for them to develop vertical apps on every portal and platform vendor in the vain hopes of reaching a sufficiently large audience. This is especially the case for providers without preexisting audio content, for whom professional recording costs add yet another element of business risk.

Audio Portal Perspective. Within the context of an aggregation model, the service provider may elect to adopt an *anchor tenant* model, similar to that employed by shopping mall developers. To jumpstart the project, each end of the mall is leased to marquee department stores at heavily subsidized rates. Having secured the consumer draw by using the name brands of the large chains, the rest of the mall will attract and be populated by smaller independents and franchises, from whom the lion's share of the lease revenue will come. This may indeed be TellMe's strategy: subsidize early channels to jumpstart its core hosting and development businesses, but it could also be a way to secure exclusivity and diminish the branding potential of other audio portals. Given the general consensus that advertising-supported only models are unsustainable, it would be difficult to offer revenue terms to each and every content provider out of an advertising revenue stream, so alternate sources need to be cultivated.

Despite the attraction of the incorporating external brands, an audio service provider may determine that in-house development of content is the appropriate tactic. In this case, the provider would not be a true portal, but more of a multi-service audio provider. For example, commodity news feeds and stock quotes can be sourced from a variety of organizations, and an application can be built without any regard for branding concerns or protracted contract negotiations. This affords the service provider the most flexibility in designing the application, but the absence of content branding makes it difficult to differentiate between the various market players.

In reality, the cost of development may be very similar for a branded or unbranded application, given that there are no content providers that have undertaken the task of making their content available in VoiceXML. Every decision to include an application must be deliberated in terms of return on investment. In contrast, given that nearly all WML sites are created and maintained by the same organization that supports the HTML site, there are absolutely no costs involved in determining the menu order on a WAP service's initial screen: it's all revenue going straight to the carrier's bank. In the absence

of a peer-to-peer Voice Web, only a selected few will be able to build out "well-stocked" consumer portals.

Enterprise Perspective. Unlike sites where the content **is** the product, much of the voice industry is looking to enterprise customers as the most lucrative targets for speech-interactive telephony. In many cases, the service required is simply a better employee locator or a way to disseminate quarterly status reports to the financial community. In other cases, speech interaction is seen as a way to significantly cut costs within their existing live agent call centers, e.g. stock quotes and trading (Fidelity), airline reservations (American Airlines), etc. J. Crew, for example, would realize significant savings if it could automate part of the ordering process: item identification and selection, availability check, quick checkout with stored credit card info, etc. Another application screaming for automation is directory assistance, one that has resisted speech recognition given the nonstandard pronunciation of names and places. Regardless of the application, these organizations are less concerned with cross-platform compatibility than total cost savings, so the service provider's role is to seek out hosting and development contracts.

Chapter 11. Structural Analysis – Tools

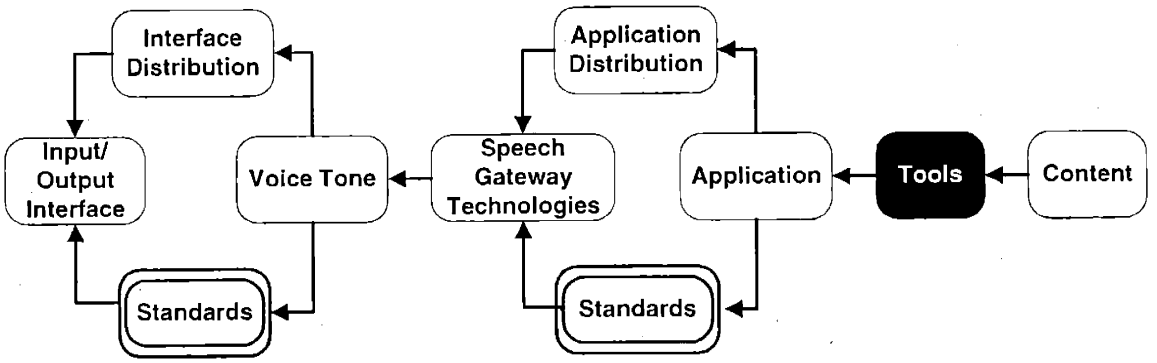

Figure 44. Market Structure (Currently on Tools)

Of all the market segments under consideration, 'Tools' seems the most unlikely candidate as the hot point for market-shaping forces. While the creation or licensing of programming tools has been part and parcel of the Internet publishing process for many years, the unique configuration of the voice architecture- *service provider-side interpretation*-provides ample opportunities for platform vendors to frustrate the "democratizing" aspect of the Voice Web. In the context of this discussion, it is important to note that the term tool applies not only to the customary software packages that facilitate the code creation and management, but also encompasses public or private libraries of portable *best practices* code and open forums for problem-sharing and discussion. So while in theory all that is required to build VoiceXML code is a simple text editor, novice and expert programmers alike would profit from the tools that are aware of the unique challenges of building speech-interactive applications, e.g. creating global commands, reprompting after timeout, disambiguation of N-best matches, etc.

11.1. Strategic Value of Tools

An artifact of the historically close relationship between the ASR engine and the business logic specification has been the ASR vendor's role in assuming the costs for tools and community development-even taking on the educator's role through their numerous outreach classes. In the days before the Voice Web was even conceivable, programmers would naturally gravitate towards these tools that simplify the coding process; however, it remains to be seen how the application community will respond to the advent of VoiceXML. As with all new standards, there is a clear tension between being the

pioneer who builds truly portable code and the cost- and time-conscious programmer that relies on existing tools that eliminate the possibility of portability.

The multiplicity of responses to this technological issue bears witness to the value of tools as a **strategic** device. The ASR vendors themselves have responded with tools that encourage the continued reliance on the proprietary dialog components. For example, Nuance's *V-Builder* tool provides an

> "... intuitive graphical interface and easy to use drag and drop environment reduces development time dramatically. V-Builder combines two emerging industry standards to accomplish this: Nuance SpeechObjects™, reusable speech components, and VoiceXML."[145]

Nuance offers the tool for free through the *Nuance Developer Network*, through which programmers are encouraged to utilize the entire library of Nuance and third-party components within the *Nuance Speech Objects Exchange*. The tool itself does not require the use of these objects, but the programmer is clearly rewarded for taking advantage of them for implementing even the simplest of dialogs. The use of Nuance's proprietary SpeechObjects does not violate the spirit of the VoiceXML specification itself (the <object> tag was ostensibly included for this very purpose), but it certainly breeches portability requirements for building a Voice Web application. The technological factors surrounding the use of these components will be discussed at length in the *Application-Speech Gateway Standards* section, but the general tension between portability and ease of code production should already be apparent. Since VoiceXML and similar efforts further the commoditization of the ASR industry, this type of response is not unexpected.

ASR vendors are not alone in their quest to preserve or extend market share through tools development. Notably, the recent deluge of speech gateway vendors on the market has witnessed the formation of many platform-specific communities. For example, Tellme Studio provides free resources (e.g. tutorials, grammars, prototyping services, etc.) for developing VoiceXML applications on the Tellme platform, and many of its competitors have followed suit with their own developer resource centers.[146] The common refrain from these platform providers has been the assurance of VoiceXML compliance while

---

[145] See http://www.nuance.com/products/vbuilder.html.

[146] See TellMe Studio (http://studio.tellme.com). Other examples include: OracleMobile Online Studio (http://studio.oraclemobile.com), Informio Developer Network, BeVocal Café (http://cafe.bevocal.com), Motorola Applications Global Network (MAGNET, http://www.motorola.com/developers/wireless/), IBM WebSphere Studio (http://www-4.ibm.com/software/webservers/studio/), VoiceGenie Developers Workshop (http://developer.voicegenie.com/).

simultaneously encouraging platform-dependence. To varying degrees, each of these platform vendors must contend with both the economic reality of promoting its own platform and the technology reality of the perceived limitations of VoiceXML itself, so all have introduced elements to their programming environments that would prevent true code portability. It must be appreciated that the moniker "VoiceXML-compatible" implies little in this nascent market, as current implementations have taken great liberties in forgoing and augmenting elements of the canonical specification. These "free" resources are not acts of altruism from the gateway community, but a strategic decision to capture market share through subsidizing a loss-leader programming assistance business.

As VoiceXML and other relevant standards are adopted by the W3C, the market should witness the disintegration of the tools industry from the ASR and platform vendors and the emergence of tools-focused companies. The world of HTML publishing is driven by a truly competitive field of tools vendors (e.g. Allaire's Cold Fusion, Macromedia'a Dreamweaver, Adobe's GoLive) without vested interests in any particular platform. In fact, unlike the consumer market, the HTML development community generally eschews those tools that perpetuate platform dominance-witness the popularity of Linux, Apache, and other open-source efforts in the XML publishing industries. If the HTML community is a precursor of things to come in the VoiceXML space, then tools will be extricated from the parochial interests of the ASR and gateway industries. This segmentation, however, will result only as the voice market grows sufficiently large to warrant tools-focused firms-a process that is far from secure given the state of standardization in the voice industry and the consequently dubious emergence of the Voice Web.

Chapter 12. Structural Analysis – Application Distribution

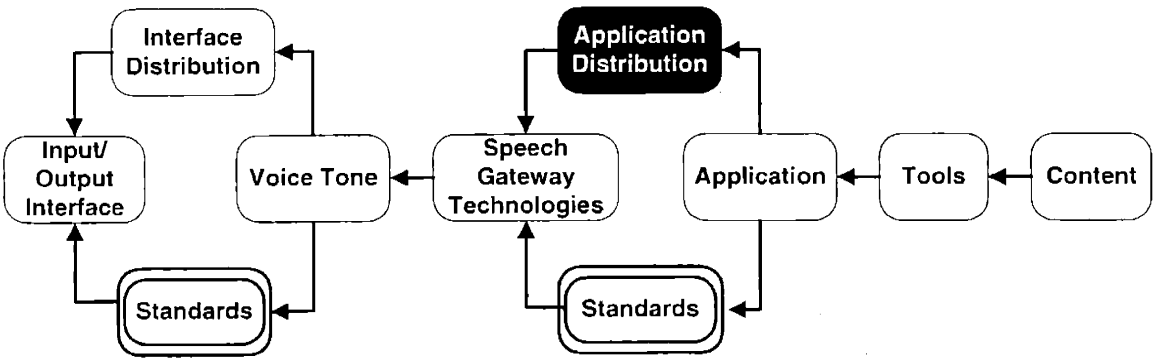

Figure 45. Market Structure (Currently on Application Distribution)

Regardless of whether the vertically-integrated gateway vendor/voice tone provider or the content provider itself is the one responsible for the creation of the VoiceXML application, the challenge of the distribution of this application remains. The consumer-oriented Audio Portal and Voice Web visions of the market both depend on the applicability of branded content already available on the public Internet. While there are certainly examples of content distribution that utilize proprietary networks to server data (e.g. Associated Press' audio feeds over frame relay and satellite networks), these instances are increasingly infrequent. Most of the content on the Internet is already engaged in HTML publishing and ready for delivery to users, so the goal is to encourage those same organizations to produce a parallel VoiceXML site, or at the very least make their content available to hosting companies or audio portal firms across the Internet.

12.1. Technology Solutions to Human Factors Issues

Given that VoiceXML is designed to take advantage of the same protocols (e.g. HTTP) and networks (e.g. Internet) employed by HTML, the challenge of application distribution is not rooted in technology *per se*, but is, in fact, driven by the *human factors* considerations of voice applications generally. Whereas human beings can synthesize and act on many parallel sources of visual information, human hearing is effectively limited to engaging a single spoken source at any given moment. So even though a web page make take considerable time to download an HTML page in its entirety, individual page elements will randomly come into focus and thus engage the user almost immediately-in other words, order does not matter greatly in visual layouts. Conversely, since humans are limited to one-dimensional spoken output,

order clearly matters and thus creates potential latency concerns-in this case, a single remote or high-latency element at the top of the VoiceXML page has the potential to hold hostage the playback of the rest of the document. Another consideration are the service expectations affirmed by decades of traditional telephony: callers are intolerant of interruptions in service, audio fidelity degradation, and simply being put on hold for a live agent. Most phonecasting consumers will not indulge the variable performance and reliability that is par for the course on the HTML Web.

While this distribution problem is firmly rooted in human factors, the technology architecture must rise to the challenge of eliminating these latency concerns. As detailed in the Technology Chapter, the most common approach for pre-recorded content is to employ content delivery servers that proactively cache static and streaming audio content[147], permitting the speech gateway to pull the content from across an unloaded segment in the Voice Tone intranet. In addition to fulfilling the core mission of low-latency audio playback, the caching schema reduces the overall bandwidth requirements for connection to the greater Internet, as well as eliminating the need for the content/application provider sites to upgrade their serving capacity to meet demand. These caching efficiencies can be further increased by encouraging the content and application providers to adopt technologies (albeit proprietary and vendor-specific) that automatically alert and propagate content to the VTPs, thus avoiding the need for frequent polling.

While the caching approach works well for pre-recorded content and "fixed" applications (static VoiceXML, grammars, etc.), the proxy serving technology affords little with respect to live content and dynamic dialog code. For example, a US-based consumer may want to listen to a live RealAudio feed of her favorite radio station in Paris. An even larger problem is the *just-in-time* (JIT)or *dynamic* generation of VoiceXML, such as a weather update customized for a particular caller's geographic preference: how can the pages be cached if they don't even exist until they are requested?

[147] The reader should recall that there are only a handful of vendors that support the capability of caching and re-serving streaming audio files in RA and ASF format.

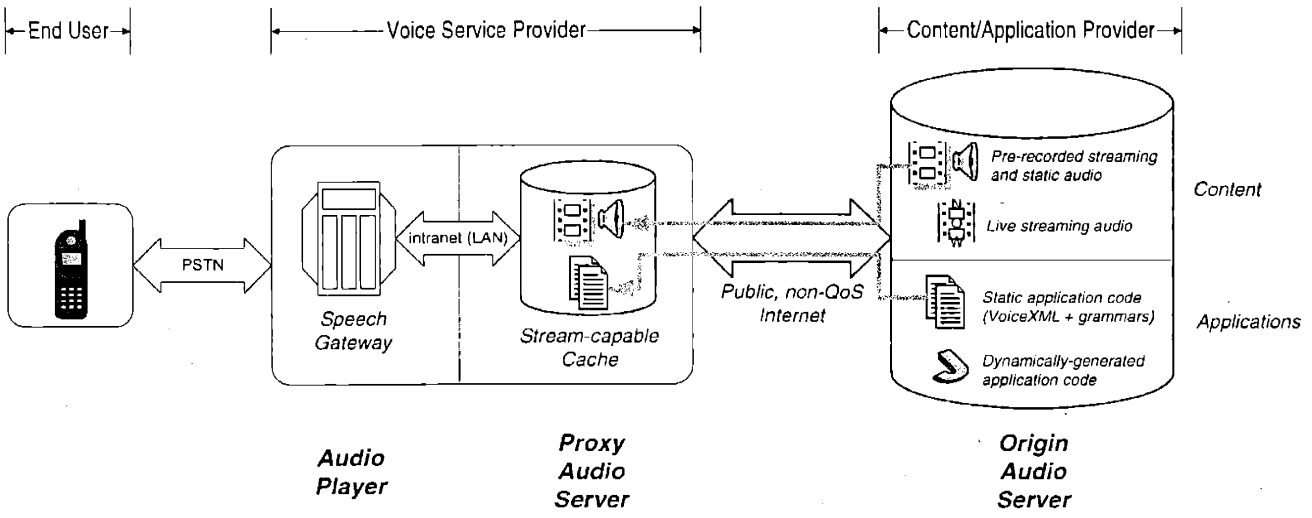

Figure 46. Data Types that Can and Cannot be Cached

Of the broad classifications for content and applications, a stream-capable caching system can service pre-recorded audio content (regardless of format) and static documents (e.g. markup, grammar, etc..) Live content can certainly be streamed through the proxy cache, such that subsequent accesses to this stream will occur locally, but this does nothing to improve the quality of experience for the subscriber to the live stream. While dynamically-generated XML and grammar files can also be stored within the cache, their value is short-lived since they are generated for a certain user and usage context-imagine the utility of caching a web page that represents step 2 of 6 in a stock purchase for a certain user. As consumers demand more personalization of their service options, dynamic markup language generation will become the standard practice for voice site development, so the cache-only schema clearly fails to address this problem.

12.1.1. Colocation

Perhaps the most obvious solution is to simply **colocate** the entire voice site within the walls of a VTP hosting firm. Back-end integration with external data sources may still be problematic, especially if the host and content/application firms are geographically remote; however, there are a number of technologies (e.g. point-to-point frame relay, satellite) that could be used to overcome these limitations. This hosting arrangement is typical in the voice industry, and adequately addresses the need of the application provider for the following two cases:

1. The application provider will offer its content only through its own phone number

2. The application provider will offer its content only through an exclusive arrangement with a particular audio portal.

Noticeably absent is the option for the wide distribution of the application, such as throughout a Voice Web. While an enterprise-focused application will be unconcerned with this type of distribution, the general consumer content industry will not invest in building voice applications with such limited scope. The colocated host can certainly serve the application with high quality, but how is the content or application conveyed to another host? Colocation is a point solution and does not provide a systematic approach to deploying a **single** voice site to **multiple** VTP locations.

12.1.2. Stream Delay

During the startup of a stream playback over the Internet, the HTML web user is accustomed to an initial "buffering" delay to smooth out the anticipated irregularities in the torrent of incoming packets. If the RealAudio Player could anticipate which stream the user will select, it could theoretically buffer the stream in the background, and then simply start a delayed playback for all callers joining the stream. Little's Theorem suggests that this solution in untenable for continuous feeds because the storage buffer (and hence delay) and broadcast length increase in direct proportion-live feeds would require infinite storage to ensure absolute quality of service over the Internet. Another problem is that this solution assumes a closed network: content can only be anticipated if it is one of the elements chosen by the voice tone provider *a priori*. A true Voice Web, like the regular HTML Web, allows ad hoc access to any site, so this buffering schema does not scale well if every stream ever made is similarly delayed. Besides, this approach affords no performance gains to dynamic dialog elements.

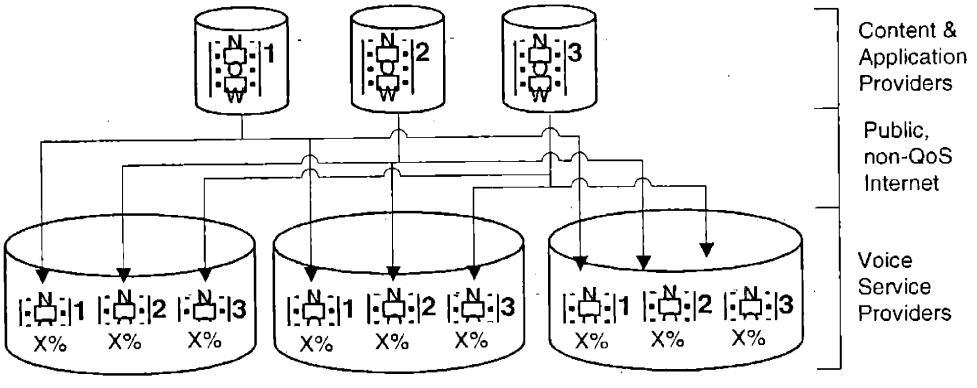

Figure 47. Stream Delay Solution

12.1.3. Application Replication

Instead of limiting the caching process to simple audio files and pages, this concept of local storage can be extended to the business logic itself. Similar to a hosting arrangement, the VSP allows provides the necessary hardware and software to run the application locally. The problem with this approach is that it creates potential costs and support complexity for the SIT network operators. For example, the VSP has to license and offer support for JavaServer, WebSphere, Perl, ASP, and any other arbitrary means by which the application provider generates XML. Further consider that these applications will likely be more complex than a trivial "Hello, world" program, so the hosting firm has to be prepared to extend its network for each accessible application. This is a sustainable arrangement only within the context of a traditional hosting agreement in which the application provider identifies a single firm to provide all of the software and hardware facilities to serve its application. Within the Voice Web context, the problem is that the application providers would ostensibly have to pay every single VSP to assume the hosting costs, a large departure from the traditional bit-conveyance role of Internet Service Providers who are paid a flat fee for access by the consumers. Needless to say, this approach is unscalable from both a business and engineering perspective: every voice site would have to be replicated at every VSP, and who would pay for it anyway? This configuration may work well for a traditional hosting arrangement, but not for a scalable Voice Web. Note that this solution also provides no performance enhancement for live streams, since the stream is being generated in real-time from a remote source.

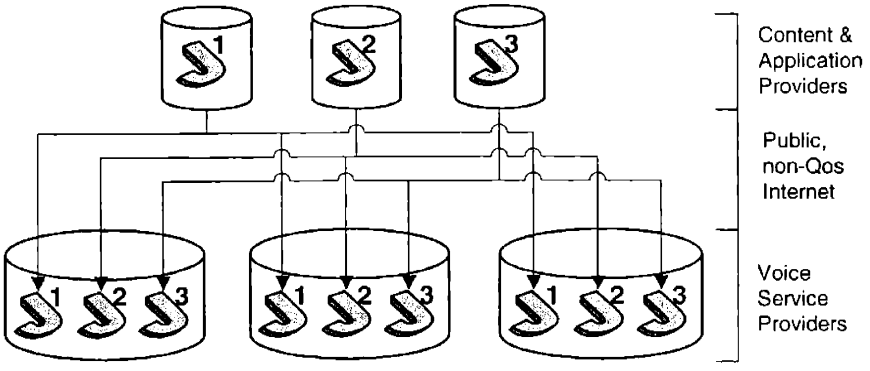

Figure 48. Application Replication Solution

12.1.4. QoS Links

One solution that would shorten the transfer latency for both live audio and JIT-XML is the use of custom Quality-of-Service-aware links between the content/application providers (whether at their own premises or at their outsourced hosting contractor) and the Voice Service Providers. There are a variety of point-to-point and virtual networking technologies that could be employed to implement these linkages, but they are significantly more expensive per bit-per-second than traditional Internet access. The advantage of this approach is that the application can be managed centrally and without concern for the hosting idiosyncrasies of each VSP In this case, even live streams are given the boost they need to overcome the limitations of the public Internet; but this solution suffers from the same scalability issues as the application replication approach above: each content/application provider would have to secure links to every single VSP.

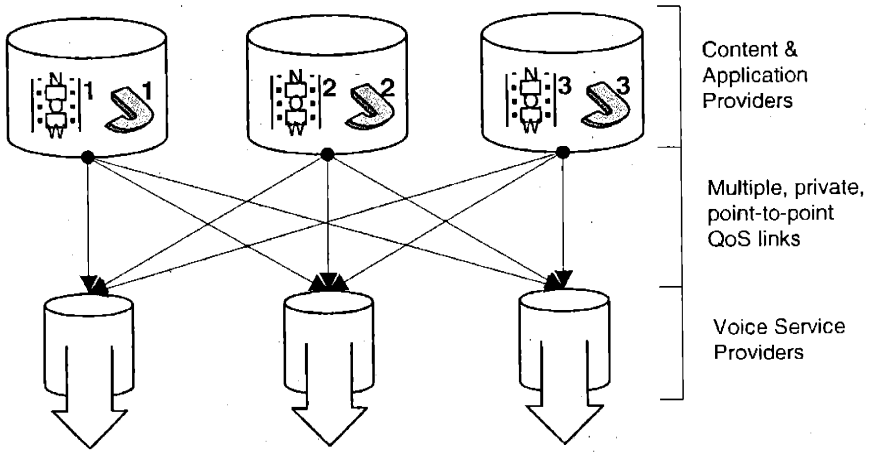

Figure 49. QoS Links Solution

12.1.5. Content Delivery Networks

The issues created by an Internet incapable of supporting QoS services have been agonized over long before the advent of phonecasting. As the demand for Internet multimedia services has grown, private *content delivery network* (CDNs) have emerged where standards have failed to materialize. CDN operators combine the traditional benefits of caching for all content and application types (including dynamic VoiceXML and live streams) by creating a private QoS network for end-to-end services in major cities. Once a CDN network is in place, any two points on the private network can communicate at speeds unachievable on the public Internet-the network is designed to minimize routing hops, employ the fastest links, etc. Consumers do not have to be similarly tethered to the CDN to experience performance gains, since the CDN operators' Tier-1 peering arrangements ensure that the distance between the CDN and a

client is minimized, to the point of only being a small percentage of the total distance, (e.g. an audio stream from San Francisco to Boston will travel over the CDN exclusively between these cities.) Finally, this content is then delivered locally by a peering partner. The content or application provider can attach to the content delivery network directly via a dedicated link, or it can simply avail itself of the hosting services of the CDN operator itself.

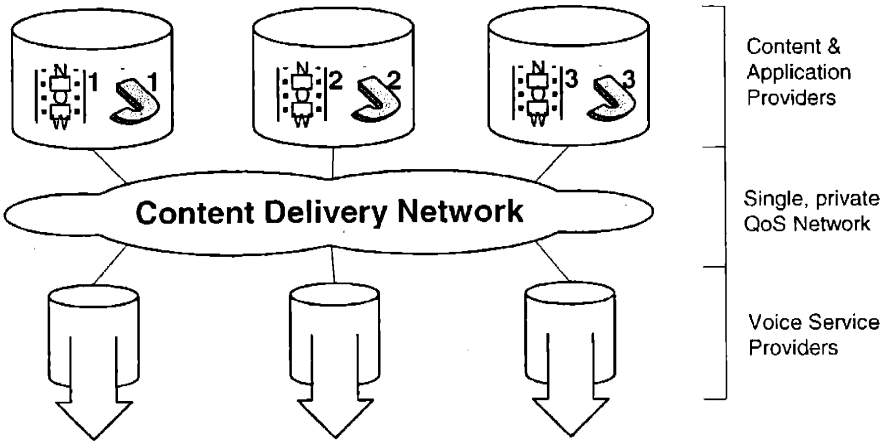

Figure 50. Content Delivery Network Solution

The problem with the CDN approach is that each organization utilizes proprietary technology and networks to convey content from end to end. The situation is not unlike the pre-AT&T era of telephony: you not only need to have a telephone, but you need to be connected to every single telephone operator. While this problem is certainly more tractable than connecting to every single content or application partner directly, the absence of peering agreements has resulted in isolated and unconnected CDNs, creating the opportunity for a single firm, Akamai, to seize the opportunity to create the positive network externality effect in AOL-like fashion.[148] Not unlike the ISP community in the early days of the post-NSF Internet[149], the rest of the CDN community banded together to form a competing content peering alliance called Content Bridge.[150] The process of content peering across organizational boundaries is structured

[148] Redherring.com article by Lisa Meyer: "Akamai is leveraging its large footprint into a critical mass of customers... Network providers are much more willing to go with Akamai, who has access to content providers. And content providers are more willing to use Akamai because of its partnerships with network providers. So the relationships build on each other." See http://www.redherring.com/investor/2000/0913/inv-akamai091300.html.

[149] After the US government stepped away from the responsibility of requiring interconnection agreements between Internet networks, large Network Service Providers threatened to withdraw from the few peering points that existed and created legitimate fears that the Internet would become an unconnected chain of IP islands. For information on the history and evolution of peering, see *Peering and Fearing* by Kenneth Neil Cukier (http://www.ksg.harvard.edu/iip/iicompol/Papers/Cukier.html) and *Scaleable Internet Interconnection Agreements and Integrated Services* by Joseph Bailey and Lee McKnight (http://ksgwww.harvard.edu/iip/cai/mcknight.html).

[150] See http://www.content-bridge.com.

closely after the bandwidth peering model of the Internet, but its success is difficult to gauge at this early stage. If the AOL-WWW model holds true for this industry, then each VTP and content provider would need to connect to only Akamai and any single Content Bridge Alliance Member to support media-grade application distribution. The economic consideration is whether the cost of the CDN services for the application community is justified by potentially expanded scope of the application's footprint.

While the author has taken every effort to remove vendor bias from the analysis, the realization of the Voice Web is firmly grounded in the reality of the market. For example, there is no hesitation in recommending the RealMedia or Advanced Streaming Formats because they almost entirely cover every audio avenue on the Internet. Despite any preference for public standards to take hold and displace the lock on the streaming audio industry, this has not happened and so it must be accepted to further another goal: the Voice Web. The same logic applies to content and application distribution: a predilection for a standards-based QoS-aware Internet does nothing to further the creation of the Voice Web, so proprietary solutions must be acknowledged when necessary and employed to the benefit of another industry. Without the end goal of creating a Voice Web, a simple collocation/hosting relationship is sufficient to deploy a single site through a single phone number since content/application redistribution is not a priority. But if the voice industry is actually sincere in its desire to eschew the walled-garden approach in favor of a Voice Web, content delivery networks are the only viable means by which to convey dynamic content with high quality.

12.2. Technology Solutions to Regulatory Factors Issues

> "Your scientists were so preoccupied with whether or not they could, they didn't stop to think if they should."
>
> Character of Ian Malcolm, Movie *Jurassic Park*

Even though the previous section has argued (hopefully) persuasively for the inclusion of content delivery networks in any viable, sustainable vision of a Voice Web, there are factors beyond consumer expectations that must be considered before pronouncing the matter closed. Any technological solution must not only address the consumer *human factors* expectations, but also preserve the *regulatory factors* expectations of the content producers on the other end of the value chain.

Conventional caching solutions are generally employed for the purpose of performance enhancement of **non-revenue objects**. For example, an ISP may choose to cache the start pages for Yahoo and other popular HTML Web destinations to reduce the likelihood of bottlenecks in its network-Yahoo would certainly not decry this practice since its subscribers would appreciate the performance enhancement. However, consider the case of redistribution of licensable content, such as television and radio shows: a company that owns the copyrights to a particular show must also have mechanisms of tracking its content to exact the appropriate amount of royalties or establish licensing rates. These revenues are the lifeblood of these content agencies, particularly those that focus exclusively on wholesaling content and eschew the XML publishing process entirely, thus they would never consent to a technology architecture that caches their content and redistributes freely without a corresponding tracking engine to manage the licensing issues. This regulatory issue is not limited to entire shows, since even the opening jingles to shows can be covered by licenses that narrowly define the permissible usage.

For several decades and long before the commercial Internet, the two major organizations that have represented artists and authors are Broadcast Music Incorporated (BMI)[151] and the American Society of Authors, Composers, and Publishers (ASCAP.)[152] They provide clearinghouses for licensors and licensees to meet and negotiate licensing rates for all type of copyrightable content, typically for the traditional broadcast industries. However, these and similar organizations have been vigilant in protecting their members' copyrights in the face of duplication and redistribution technologies (i.e. caching) on the Internet, and they have been quick to anticipate and present licensing options to HTML sites that build their revenue base around content distribution. Consider the following Webcasting license offered by BMI:

> "Many web sites base their business on the use of music, while others offer it as an adjunct or compliment to the rest of the content on their site. BMI offers several licensing options in recognition of the diverse and evolving nature of business on the web... The standard web site license is generally for commercial entities that generate revenues from the operation of the web site and offers the flexibility to choose from two financial calculations to determine your license fee based on the nature of your web site as follows: The Gross Revenue Calculation can be used if you are using music as a primary feature on your web site. The Music Area Calculation allows

[151] See http://www.bmi.com/.
[152] See http://www.ascap.com/.

you to reduce the revenues subject to fee by factoring the traffic to pages with music in relation to your total web site traffic."[153]

While there is no specific mention of VoiceXML sites in the licensing statement above, phonecasting activities are clearly covered under the more general electronic distribution rules of the United States Copyright Office[154] and the World Intellectual Property Organization.[155]

As a business model that implicitly relies on the ability to redistribute branded content, the implications for phonecasting are twofold. First, any phonecasting agency that believes that it can simply point to content already on the web and redistribute without explicit licensing agreements will be pursued vigorously by the content industry in general. There is a clear distinction between technologies that simply aid visually-impaired users in surfing the existing visual web and a phonecasting business model that relies on branded audio to drive adverting and other revenues. The ease and availability of audio on the Internet does not abrogate the trademark and copyrights of the content creators. Second, given that licensing is a required function of any portal that relies on branded content, there must be mechanisms in place to reconcile the conflict between the "cache-once/redistribute-many" performance philosophy and the "track-every-download" copyright policy.

Recognizing these regulatory factors early on, the content delivery network operators and equipment vendors have embraced the this notion of digital rights management and include it within their core offerings, e.g. Akamai Digital Parcel Service[156], RealServer Proxy Accounting Connections,[157] etc. These solutions enforce a variety of content policies that are defined by the copyright holders, e.g. members only content, per-click fee, per subscriber flat fee, etc. While there are no open standards that allow digital rights management across CDNs currently, the emergence of Content Bridge CDN alliance perhaps signals a movement towards ubiquitous rights management and is not a gating factor to building a voice portal. In summary, the CDN is the right technology choice to meet the *human factors* needs of consumers and the *regulatory concerns* of the content industry.

---

[153] See http://repertoire.bmi.com/iama/webcaster/webans1.asp.
[154] See http://www.loc.gov/copyright/.
[155] See http://www.wipo.org/.
[156] See http://www.akamai.com/html/en/sv/dps_over.html.
[157] See http://www.realnetworks.com/products/servers/proxy/.

Chapter 13. Structural Analysis – Application ↔ Speech Gateway Standards

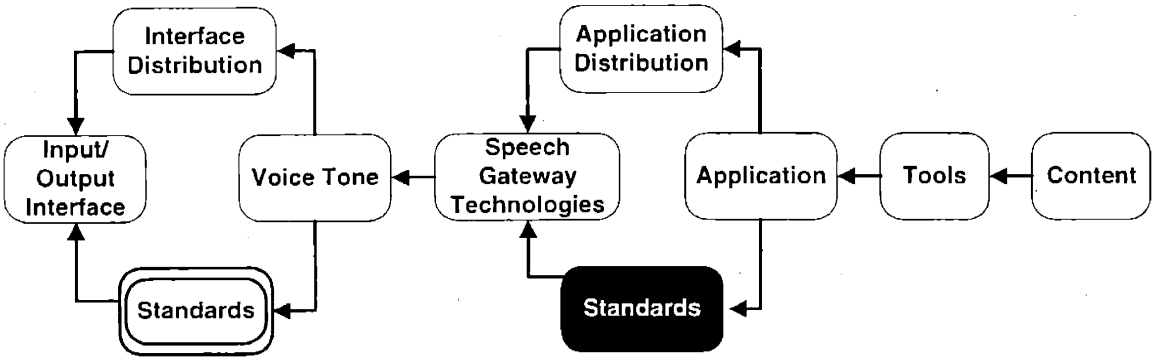

Figure 51. Market Structure (Currently on Application – Speech Gateway Standards)

As the Technology Primer demonstrated, the evolution of the voice industry has been largely restricted to improvements in technological performance-the industry structure itself has changed very little. The introduction and quick adoption of VoiceXML as the standard dialog markup language has afforded the industry's first opportunity to vertically disintegrate the speech gateway and application creation market segments. This does not imply that speech gateway vendors alone were capable of implementing applications on their platforms (though this is not an uncommon practice), but it does herald two significant trends in the industry:

- **Application creation will be put back into the hands of the content publishing firms**: "VoiceXML's main goal is to bring the full power of web development and content delivery to voice response applications... VoiceXML shields application authors from low-level, and platform-specific details."[158]

- **Applications will be portable to any VoiceXML-compliant gateway platform**: "Promotes service portability across implementation platforms. VoiceXML is a common language for content providers, tool providers, and platform providers."[159]

[158] Section 2.2 of *VoiceXML Specification*.
[159] *Ibid.*

It would appear patently obvious that an open standard such as this encourages large-scale adoption and general market efficiency, but the list of standards that have thrived in closed environments is lengthy: ATM, Acrobat, RealAudio, Word, ZIP, etc. Particularly considering the voice market's long tradition of vertical integration and oligopoly, the popularization of VoiceXML is only one piece of the standards puzzle.

Most languages and document standards arise from necessity, e.g. an ASR vendor needs a way for programmers to communicate with is engine. In those cases when an existing standard, particularly one not beholden to a hostile firm, meets the design criteria, firms will consider adopting it. With respect to the voice space and specifically the Voice Web concept, document and protocol formats are not driven by cost but by *network externalities*-the recognition that the failure to standardize certain parts of the machinery will result in a smaller pie or none at all. Firms are not suspending their desire to maximize their own profit, but they will suspend competition at select points in the value chain if it is the keystone for a much larger market. The initial wave of support behind VoiceXML is strong signal that the dialog format is one of these key pieces. In short, the question is not whether languages or protocols are expensive to create, but what is the opportunity cost of failing to create them?

General economic factors aside, the most profitable analysis of VoiceXML and related standardization efforts is firmly rooted in technological considerations. The Technology Chapter outlined characteristics of a language that is well suited for the phonecasting industry in particular-interpretable, portable, open, XML-based, and useful. From a high-level view, it appears as though VoiceXML scores well on all accounts, but the proverbial devil is in the details (you used this expression before, page 98). For example, *portability* is compromised both by elements included within the standard and those that the designers failed to include: errors of *commission* and *omission*. The overarching *utility* of the standard is also impugned by limitations on grammars visible across multiple documents, for example. Finally, the language will be examined from the more demanding viewpoint of the Voice Web model to see where it either hinders or stimulates the evolution of the industry towards this goal.[160]

13.1. Portability: <script> and <object> tags

[160] On a positive note, it seems clear that VoiceXML does indeed satisfy the requirements of being both an *open standard* and an *XML-schema*.

While the unification of the industry around a single speech application language is a notable event in and of itself, the more important observation is that it has taken the form of an XML schema instead of the traditional compiled language. The programming shift from a procedural to a declarative language allows this new language to leverage the tools, protocols, and platforms already employed to drive the HTML and WML industries; however, there continue to be lingering concerns that an XML-based language lacks the sensitivity to timing that is essential to deploying a viable speech application.

There is a clear antecedent to the claim that VoiceXML alone cannot implement every conceivable speech application, or conversely, that a programmer might have to exert a Herculean effort to implement a program using only VoiceXML Looking to the example of the visual web, no amount of HTML is capable of reproducing a word processing application with a real-time graphical user interface and with a rich set of functionality (e.g. Microsoft Word.) The popularity of Macromedia's Flash and Shockwave plug-ins testifies to HTML's inability to support real-time interactive website navigation and animation. While the majority of applications on the visual web can be constructed within the familiar markup schema, there is no denial that real-time interactivity is a challenge for the XML programmer.

Whether the motivation is the latency of the link between client and server, or simply a desired feature beyond the scope of markup, visual web developers saw the need for *client-side*[161] *scripting languages* that would retain their link to the server but still offer the end user the real-time interactivity required for the application. The two major HTML browser vendors each came up with their own proprietary scripting solutions: Microsoft's JScript and Netscape's JavaScript. While programmers were now armed with more powerful tools with which to build web applications, they had to contend with the fact that the limited (yet ubiquitous) HTML web seemed preferable to a world now segmented into two irreconcilable halves. In an attempt to forestall the fracture of the entire web community, these two firms enlisted the assistance of the *European Computer Manufacturers Association* (ECMA)[162] to create a single scripting standard that would be adopted by all browsers. The result was ECMAScript[163], which is not a scripting language itself, but a specification to which all web scripting languages should adhere. Specifically, the ECMAScript standard describes the core language with data and objects (e.g. numbers, strings, arrays, date, function), but eschews objects specific to each browser (e.g. document, window.) JScript and

[161] Remember that *client-side* scripting actually occurs on the *service provider* premises since the VoiceXML interpreter resides on the speech gateway.
[162] From ECMA's website: "For over thirty years ECMA has actively contributed to world-wide standardization in information technology and telecommunications. More than 270 ECMA Standards and 70 Technical Reports of high quality have been published." See http://www.ecma.ch/.
[163] The ECMAScript specification can be downloaded at ftp://ftp.ecma.ch/ecma-st/Ecma-262.pdf.

JavaScript (though to a lesser degree) currently conform to these specifications in that all of ECMAScript is implemented, but they both continue to offer features and functionality above and beyond the standard. The key to cross-browser compatibility is to restrict one's use of JScript or JavaScript to the ECMAScript specification, and avoid the proprietary elements of each language (even though they can be very useful from the programmer's perspective.)

Anticipating the need for client-side scripting as well as avoiding a reprise of the scripting "wars," the VoiceXML Forum created the **<script>** tag with the condition that ECMAScript support is a requirement of all interpreters. In addition, the ECMAScript conventions for declaring variables and expressions are adopted throughout the VoiceXML specification, further solidifying support for this scripting standard. The explicit inclusion of ECMAScript support within VoiceXML affords a powerful and flexible environment within which to create elaborate speech applications. For example, TellMe's well-regarded consumer portal is implemented using only VoiceXML 1.0 and JavaScript 1.2[164], and the VoiceGenie VoiceXML Gateway relies entirely on VoiceXML and ECMAScript for dialog specification.

The added flexibility of a scripting language certainly simplifies problems such as those of complex input interactions and formatting results, but it leaves unanswered the issue of creating grammars to recognize the inputs in the first place. Recalling the grammar discussion from the Technology Chapter, a task as deceptively simple as recovering a string of digits from a spoken fragment is a challenging programming task. Again, the creators of VoiceXML anticipated the difficulty that novice programmers would face in performing even the simplest of input tasks, so it required support for the following *builtin grammars*: Boolean, date, digits, currency, number, phone, and time. The programmer could certainly feel free to create her own grammar for any purpose, including an improved version of the platform provider's default implementation-so long as the added grammars "traveled" with the VoiceXML document, the target VoiceXML interpreter instructs the ASR engine accordingly.

This trio of VoiceXML, ECMAScript, and builtin but also extensible grammars provides a strong multi-platform framework upon which to build most any type of speech application. Traditionally, the combined functionality of this trio was assumed by the reusable dialog component: a function call that would abstract away the logic and grammar complexity involved in processing a speech input. Each ASR vendor provides its own proprietary, incompatible set of reusable components to simplify the programming task, e.g. Nuance SpeechObjects and SpeechWorks DialogModules. For example, Nuance's

[164] From looking at examples of code on its developer website (http://studio.tellme.com/), it appears that TellMe is indeed limiting its use of JavaScript to only those elements compliant with the ECMAScript standard.

SOCreditCardInfo component obtains the credit card type, card number, and expiration date, combining elements of both logic flow and grammar design. Even armed with ECMAScript and the builtin grammars, the programmer would be reticent to eschew a ready-made object if it were available. Even though the major vendors provide similar components, they can differ in both number and style, and they can optionally take inputs to further vary the functional flow. Even beyond speech recognition, platform vendors may want to make visible other aspects of their systems, such as speaker verification and advanced telephony functions. The VoiceXML creators recognized the desire to retain access to these and other platform-specific elements and codified their access through the **<object>** tag. Given the inherent flexibility of using the tag, it is "standardized" to the following extent: the return value of the execution is an ECMAScript object, and support for the tag is required (regardless of whether any platform-specific objects are actually created.)

From a high-level viewpoint, the <script> and <object> tags are similar in that they serve to extend the core functionality of the basic markup language itself; however, their usage has implications for the creation of the VoiceWeb. **Specifically, the object tag's inclusion in the standard extends functionality for a single platform, but destroys portability across platforms.** For example, if a programmer relies extensively on the VoiceXML object tag to access Nuance SpeechObjects that tackle tough inputs, then that code is not portable to any platform that does not support SpeechObjects, e.g. TellMe uses Nuance's ASR but chooses not to encourage their use by supporting them through the object tag, Nortel's Audio Browser uses a SpeechWorks ASR engine and therefore has no ability to execute the SpeechObject. Since even simple inputs can be considered complex-timeouts, disambiguation, or confirm-and-correct cycle- many programmers would be encouraged to use VoiceXML as a thin placeholder for the dialog, scripting, and grammar logic contained with these dialog components, a practice that several platform providers intentionally facilitate. For example, the Nuance V-Builder tool encourages the use of SpeechObjects to perform even the simplest of dialogs within VoiceXML. During its AVIOS 2000 presentation, BeVocal made no secret of its belief that Nuance SpeechObjects were the only viable vehicle, at least in the short-term, for delivering phonecasting applications. The BeVocal interpreter not only supports Nuance SpeechObjects, but also licenses the very same objects it uses itself to deploy the consumer portal, e.g. VocalNews, VocalTraffic, VocalWeather, etc.

There is no intent vilify Nuance and BeVocal for promoting the use of platform-specific components, nor even to impugn the utility of the components themselves. In fact, if the application provider expresses a clear disinterest in making its content and services available through the platform-independent Voice Web, then the programmers for that application should feel free to use any tools at their disposal to create

the application, including dispensing with VoiceXML altogether in favor of the more familiar compiled, proprietary speech language. However, if the application or content provider does indeed have a vision for distributing its content throughout the Voice Web just like its HTML and WML sites, then these components must be strictly avoided, even at the cost of some functionality. Regardless of the utility or sentiment driving the use of the object tag, its usage is antithetical to the building of a Voice Web.

The programmer faces the classic tradeoff of utility versus portability that so often arises in the application of XML. Suggested solutions to address this compromise include:

- **Extend the Standard**. If certain grammars or functions are commonly encountered but not covered by the standard built-ins, then the W3C/VoiceXML Forum should consider including them in successive releases of the standard. However, this is as slow process at best, continually lagging behind the needs of the market.

- **Tool and Library Growth**. As with many XML industries, the community of programmers will stimulate the creation of reusable dialogs that rely on **only** VoiceXML, ECMAScript, and grammars. Consider the tools and code libraries available to web programmers only 4 years ago in relation to those currently employed: a novice can use a graphical program to create a website without knowing a single tag of HTML. Some will be made available to the public domain, and others will be commercialized, but the end result is a rich platform-agnostic foundation for building speech applications without resorting to object tags.

- **Plug-Ins**. In those cases where no amount of tools or creativity will suffice (e.g. implementing Word in HTML), special objects can be created, provided that its creators make a concerted effort to popularize and port the necessary plug-ins to the most popular platform vendors. For example, Macromedia makes a concerted effort to include its Flash technology as an integral component of every major browser, as does RealNetworks for its RealAudio Player. Generally, this approach is discouraged because it violates the spirit of standardization and it would simply present the user with an error if she happened to be using a system that didn't support a random plug-in (in this case, she doesn't have the standard option of downloading and installing a new plug-in on someone else's interpreter.)

In practice, this matter of component reuse is difficult to surmount given the tendency of platform providers to leverage these proprietary elements as differentiators. For example, given that continued

reliance on SpeechObjects furthers its ASR licensing revenues, Nuance would not likely discourage programmers from using them in the name of cross-platform portability, especially since it would be commoditizing itself as an ASR engine and facilitate code movement to other ASR vendors. Of course, this would be a short-sighted perspective, given the revenue potential for ASR sales in a Voice Web market rather than a market comprised of walled gardens, but the tension exists and must be appreciated by those pushing for standardization.

Even TellMe, an organization ostensibly dedicated to furthering the Voice Web, creates cross-platform issues through its *Intrinsic Grammars*: a set of grammars that TellMe has created to assist the programmer in enabling complex inputs without resorting to an object call.[165] This certainly embodies the spirit of true Voice Web programming, since all aspects of the code utilize only the VoiceXML/ECMAScript/grammar trio and could be executed on any VoiceXML-compliant platform; however, the **source code** for these grammars is usually kept hidden from the user, and there is no indication that one could simply license these grammars for use on a competing platform. Again, this assumes no malice on the part of TellMe but a simple economic decision: What would TellMe gain from opening these grammars, especially if it would encourage the use of a competing platform? Whether a programmer relies on ASR engine-specific objects or cross-platform elements whose source code is hidden, the outcome is potentially the same: the failure of the Voice Web to materialize. The author would suggest that the accolades garnered by TellMe's portal prove that a programmer need not resort to ASR-specific code to implement a compelling user experience, thus the application of best-practices ECMAScripts and grammars are clearly the best option for extending the utility of VoiceXML; however, they need to come from a community not vested in the furtherance of a particular platform, whether that platform is an ASR engine or the speech gateway itself.

As part of its general Voice Browser Activities, the W3C is working on two projects that could be used to further discourage the use of proprietary dialog components through the object tag. First, the Natural Language Semantics Markup Language[166] provides a data interchange format that focuses on representing detailed semantic information from a single user utterance, allowing the engine to perform more sophisticated pattern matching than traditional ASR. Since many proprietary dialog components are designed with natural language implementation in mind, the inclusion of this specification would ostensibly reduce the need for these objects. Second, the Reusable Dialog Requirements Subgroup was created with the following mission of developing a "specification for reusable dialog components within

[165] See http://studio.tellme.com/library/grammar/ for a description of these intrinsic grammars.
[166] See http://www.w3.org/TR/nl-spec/.

the context of an overall specification for a markup language."[167] In a sense, this is an effort to standardize the interface to plug-ins; however, the current working draft indicates that the effort is less ambitious that the title would suggest:

> "Although desirable to standardize the interface to all dialog components, this standardization is impractical for many dialogs. In order to standardize the interface, one would have to standardize the call flow, since the specifics of the call flow determine the parameters that can be configured. If this document attempts to standardize the call flow (and hence interface) for more complex and debatable dialog components, the resulting standard components are likely to contain only the lowest common denominator of functionality and therefore be of limited usefulness. Even the control flows for such common tasks as acquiring telephone numbers and postal codes can differ from one application and vendor to another."[168]

The motivation for this discussion is the unwavering need for platform interoperability in support of the VoiceWeb, and the necessary compromises that such a commitment entails, especially in the short term when best-practices tools and libraries are simply not available. While the components summoned through the object can have obvious utility for the programmer, they can be largely reproduced with the more basic VoiceXML/ECMAScript/grammar components already available without incurring the cost of platform specificity.

13.2. Portability: Grammar and Synthesis Languages

Implicit in the discussion above is the absolute portability of the code to any VoiceXML-compliant platform. Both the VoiceXML markup language itself and ECMAScript are well-defined, but the matter of grammar and synthesis languages bears further examination. The grammar language is intimately tied to the functionality of the ASR engine, so the role of format and syntax definition has traditionally fallen to the ASR vendors themselves; similar logic holds for the tight relationship between a synthesis language and its corresponding TTS engine. While the utility of TTS within a Web well stocked with professionally recorded and branded audio is questionable, there is no quarter for diminishing the importance of the grammar language-not simply a laundry list of words and phrases, it provides the low-level logic to the ASR engine to recover nontrivial inputs from speech fragments.

---

[167] See http://www.w3.org/TR/reusable-dialog-reqs.
[168] *Ibid.*

While the advent of VoiceXML signaled a renewed industry commitment to cross-platform portability, the reader has already witnessed how the use of the object tag can eliminate that utility, and the grammar and synthesis languages provide yet another example of this insidious, creeping incompatibility. Returning to the example of TellMe's commitment to the standardization process, suppose that an executive-level decision was made to earn additional revenues by licensing the source code to its entire cadre of grammars and code libraries. The problem is that portability is still restricted by TellMe's reliance on the Nuance Grammar Specification Language (GSL), an artifact of its adoption of the Nuance ASR engine for its platform. BeVocal is also similar tied to GSL because of the Nuance ASR, and one would expect the same platform incompatibility for ASR-specific grammar specifications. So despite the best intentions of the VoiceXML Forum and even if TellMe hypothetically opened up its code repositories, this still amounts to dialog logic that is hostage to the platforms themselves, which is exactly what VoiceXML was ostensibly designed to prevent.

Clearly, VoiceXML is not enough, a fact that the W3C has both anticipated and responded to. In addition to accepting VoiceXML 1.0 as the basis for DialogML, the W3C has created working drafts of grammar (Speech Recognition Grammar) and synthesis (Speech Synthesis Markup Language) specifications. The SRG and SSML specifications borrow heavily from Sun's JSpeech Grammar Format[169] and JSpeech Markup Language[170] specifications, respectively, both of which where the result of previous industry collaboration and were submitted by Sun to the W3C for consideration.[171]

While the W3C has clearly identified the need for standards beyond the dialog language itself, this process may not react quickly enough within the formative stages of the voice industry. Other than the fact that DialogML will be based on VoiceXML 1.0, there is neither working draft nor any planned date for the release of the specification recommendation. The W3C grammar and synthesis specifications are currently in working draft form, with recommendation dates set for November 2001. While the entire W3C organization is understandably in a constant mode of catching up to industry practices and then codifying them for broader use on the web, it must have the sense of urgency to forestall the walled garden effect that is **already** occurring in the absence of these standards. Just as the 0.9 release of the VoiceXML specification spurred developers to create interpreters in anticipation of the 1.0 version, the

[169] See http://www.w3.org/TR/jsgf/
[170] See http://www.w3.org/TR/jsml/
[171] For a general summary of the W3C's efforts in the voice arena, see http://www.w3.org/Voice.

hope is that platform and ASR vendors will proactively incorporate the spirit of the W3C Speech Recognition Grammar ahead of its official release.[172]

13.3. Portability: Shadow Variables

While certainly not as notable as the platform portability issues created by proprietary dialog components, grammar formats, and synthesis languages, there are still further elements within the VoiceXML standard that would work against the formation of the VoiceWeb. The first example is the commonly desired ASR feature of N-best matching: instead of returning a single value and confidence score for that match, the ASR engine should be able to return an array of up to N elements and confidence scores for a single match. For example, if a large directory assistance application has to deal with many similar-sounding names, N-best matching allows the user to further disambiguate the input given the top N choices without resorting to a live agent. Several vendors have suggested extending the *confidence* and *utterance shadow variables*[173] to create support for N-best matching, with the understanding that any VoiceXML code that utilizes it will naturally be unsuitable for interpretation on any other platform. Another suggestion for N-best matching is resorting to the more familiar reusable dialog component, but this also incurs the penalty of platform restriction. Until the W3C explicitly includes N-best matching in the DialogML 1.0 release, programmers desiring platform flexibility may simply have to do without this important feature.

13.4. Portability: <property> tag

The <property> tag is perhaps the most "forthright" of the platform-specific elements of VoiceXML in that its only role is to control proprietary parameters. They are typically employed to modify certain basic behaviors of the speech gateway-recognition- vs. energy-based barge-in, timeout intervals, caching policies-and they can be defined globally for the entire session or for form-level granularity. While these tags do not typically occur with great regularity in VoiceXML code, their usage is entirely driven by the

[172] For example, the VoiceGenie VoiceXML Gateway supports the Nuance and Watson grammar formats, but it also anticipates the W3C Speech Recognition Grammar by supporting the *Augmented Backus-Naur Form* (ABNF) through real-time conversion to Nuance GSL. This prevents a hypothetical strategic move by Nuance to protect is ASR sales by refusing to support the W3C grammar standard. See: http://developer.voicegenie.com/techinfo/vg_voicexml_2.0R_v1.0.pdf.

[173] Each field item may have an associated set of variables that augment or "shadow" the basic value returned from the ASR engine. The confidence shadow variable signals a confidence score for the match, and the utterance variable represents the raw tokenization of the speech fragment. Each of these shadow variables is optional, and can be defined arbitrarily by the platform vendor.

needs of the application. While the VoiceXML standard refers the platform implementers to Sun's JSpeech API[174] for a list of "generic" platform parameters, speech gateway or ASR vendors that differentiate themselves through feature sets will not hesitate to employ this tag to access them. Fortunately, there is a clear precedent in the dynamic generation of HTML to account for the idiosyncrasies of the Internet Explorer and Netscape Navigator browsers, so XML programmers are accustomed to dealing with the occasional platform-specific but peripherally-important tag. The objection to reusable dialog components and ASR-specific grammars is that the entire dialog logic rests on parochial substrate, whereas the property tag is likely used sparsely to affect more subtle parameters of performance (and not correctness.) The property tag is a necessary element of a robust speech language, but the programmer interested broad Voice Web exposure will do well to avoid excessive use of it.

VoiceXML's *portability* across speech gateway platforms has been the focal point of the foregoing discussion, but this is not the only relevant avenue along which VoiceXML must be considered. Even in the absence of portability concerns, the language's *utility* must be examined in light of the applications that it purports to champion.

13.5. Utility: Audio

13.5.1. Audio Formats

While the mechanics audio streaming have been discussed at length in the Application Distribution Section, the matter of audio standards themselves has not, particularly from the perspective of the VoiceXML specification. Analogous to the situation of grammar formats, the specification abstains from requiring support for any particular audio format, but implicates several in an appendix. The committee members suggested 4 formats in Appendix E, each of which support static/8-kHz/8-bit/mono audio but with slightly different encodings (e.g. μ-law, A-law.) The lack of requirements for any particular audio format is consistent with the W3C's spirit of impartiality: to endorse the proprietary formats and protocols of the two major streaming audio vendors would be a serious breach of the organization's credibility.

[174] See http://www.javasoft.com/products/java-media/speech/index.html.

The practical effect of this omission is that the prerogative for selecting and integrating these streaming audio technologies falls to the gateway vendors themselves-a situation that closely parallels the evolution of the HTML industry. As the browser industry became quickly dominated by the oligopoly of Netscape Navigator and Microsoft Internet Explorer, it was easy to lobby each vendor individually to include new features, and the browsers themselves embraced the plug-in concept: allowing vendors to create custom player modules for proprietary media types. So in effect, neither Navigator nor Internet Exporer has ever explicitly supported RealAudio, for example, but RealNetworks was allowed to offer a downloadable application that could interpret and present the RealAudio stream.

The operative question is whether these are relevant precursors for the VoiceXML gateway vendors. If the application demand for streaming audio exists, then it would be difficult to imagine that the gateway industry would not respond positively to integrating these technologies. However, the chicken-and-egg paradox is operative here: applications that rely on streaming audio can't be built, but streaming technology won't be incorporated until sufficient demand results. If RealNetworks and Microsoft are serious about protecting their duopoly, it may very well be the case that they will subsidize the efforts of the gateway industry to assure compatibility. Another concern is the user interface itself: what is the audio equivalent of "spawning a new audio player window"? In the absence of audio *navigation* standards, this concept of recruiting an external "helper" application to handle an audio feed is untenable.[175]

Given that audio isn't just an important element of the voice application experience but practically the entire experience, the lack of support for streaming formats should be the subject of active discussion in the voice industry. The reality is that the speech gateway community must conform to the needs of the application providers since no amount of emphatic assertion is going to change the course of entire broadcast industry on the web. An audio portal dedicated to recruiting the largest names in branded audio will be continually rejected on the grounds that they cannot support their existing audio publication technologies.

For example, On24, the leading multimedia aggregation network for financial news and information, is built around the RealAudio and ASF formats. The author would be hesitant to suggest that they totally discard or augment their entire distribution network to satisfy the needs of a particular speech gateway vendor. Audible.com would similarly reject any proposition that it recode its 24,000 hours of content to

[175] The plug-in concept also fails on the grounds that the Voice Tone Provider would be reticent to allow each caller to customize her own VoiceXML browser, thereby compromising the scalability and reliability of the entire system.

accommodate the idiosyncratic, IVR-focused tendencies of the gateway industry. Compounding this lack of streaming support with the unclear realization of the Voice Web to justify production, it is easy to appreciate the reserve of the content industry in embracing VoiceXML.

13.5.2. Audio Navigation.

The absence of streaming audio affects the standard at a deeper level as well, because VoiceXML implicitly regards all audio as *short-play*, e.g. prompts, 20-second sound bites. Supporting the playback of streaming audio involves not just the mechanics of reconstructing the audio sample from the bitstream, but also supporting the relevant functions of *long-play* audio. For example, a caller might listen to a book during her commute to work, and she would obviously like to perform functions such as: bookmark, resume playback, control volume, fast-forward or rewind, pause-functions that are readily available on any streaming player on a desktop computer. These long-play functions require access and mobility within a single audio file, but VoiceXML's audio granularity stops at the file boundaries.

Imagine the frustration of the following user: A businessman is listening to the 10-minute CNN Headline News Update over his cellphone. He is already frustrated that he cannot simply fast-forward to the business update segment that starts 8 minutes into the file. Just before his segment begins, a person speaking next to him triggers the barge-in mechanism he is then requested to issue a new command or return to playback. When he chooses to return, he is met with the unwelcome surprise of having to start the entire segment over from the very beginning. While starting over might be fine for a 5-second prompt, it is completely unacceptable for long-play audio.

Functions like pause and rewind are easily accessed through the streaming vendor's SDK[176], the issue, however, is how to convey the user's commands to the audio player through VoiceXML. Unlike a typical ASR input that is acted on within the VoiceXML code itself or reported to an external server, the successful recognition of an audio command needs to be routed to the audio subsystem. This process could take many forms-a proprietary object call from within the VoiceXML dialog that the interpreter routes to the subsystem-but it most certainly implies some sort of code that would not be VoiceXML-compliant and therefore not portable across platforms. This would also introduce additional complexity in

[176] If the speech gateway vendor elects to forgo streaming support and develop its own audio engine, then it will also have to address the challenge of supporting these higher-level functions for all of the supported audio types. In short, simple playback may not be enough.

the browser, since it would need to preserve the state(?) between dialogs or pass audio-level session info back to the server, e.g. if a stream is paused, where should it be resumed?

The need for intra-file or intra-stream navigation introduces yet another element of concern: enforcing the *universality* of audio commands across organizational boundaries. For example, all audio streams are accessed through the appropriate vendor's Player application, which enforces universal stream navigation commands across all content and site providers. Since the Player module now resides at the Voice Tone Provider premises and the speech gateway vendor has free choice over which audio types to support, the caller is not in a position to simply "download" the Player plug-in for the VoiceXML browser. Not only does VoiceXML fail to enforce a consistent stream of navigation commands, it would also be unclear as to how to prevent VoiceXML application providers from using the same keywords for legitimate use within their own code. This question of universality touches not only audio navigation but also content navigation generally, and will be subject of further examination later in this chapter.

13.5.3. Potential W3C Audio Contribution

As of the writing of this paper, the W3C had still not released the Working Draft of its *Dialog Markup Language(DialogML)*, which would essentially correct and update the VoiceXML 1.0 specification as the starting point for the new standard. However, the W3C's Voice Browser Group did make available design requirements for DialogML, among which the following statement about output content is labeled as a "must have":

> "... the [dialog] markup [language] supports the following output features (if not already defined in the Synthesis Markup):
>
> 1. Pre-recorded audio file output
> 2. Streamed audio
> 3. Playing/synthesizing sounds such as tones and beeps
> 4. Variable level of detail control over structured text
>
> The output device generates timestamped events including error events and progress events (output started/stopped, current position.)"[177]

---

[177] Section 3.9 of Dialog Requirements for Voice Markup Languages, W3C Working Draft, 23 December 1999. See http://www.w3.org/TR/voice-dialog-reqs/.

The inclusion of the terms "streamed audio" and "timestamped progress events" are encouraging in that they suggest a recognition of the distinction between short-play and long-play files, the latter group implemented almost universally as audio streams. The reference to timestamping creates the possibility for supporting a resume function:

1. The DialogML interpreter is required to support playback of audio at a progress offset, e.g. time, bytes, pointer, etc. It is further required to report that offset if a barge-in or similar event occurs to interrupt playback.
2. The DialogML code or the document server can act on this information to support resumption of playback after interruption, bookmarking, etc.

Noticeably absent from the requirements document is any mention of creating or standardizing audio navigation commands, but this may be due to the assumption that the only implied navigation function is "start playback at offset specified," and not the pantheon of streaming functions like "rewind," "pause, "louder," etc. Also missing is any mention of supporting the RealAudio or ASF formats specifically, which is entirely in character for the impartial W3C.

13.6. Utility: Multimodality

VoiceXML's roots in the larger XML movement convey implicit support for multimodal publishing: the dynamic generation of various schemas from a common XML core. Rather than create a vertical application for each access mode, programmers are encouraged to employ style sheets and higher-level server-side scripting languages to generate the appropriate XML variant in real-time. This does not suggest that creating good voice applications is an intuitive process for an experienced HTML programmer, but it does indeed leverage the existing, familiar back-end generation and integration tools.

Implicit in the above discussion is an *asynchronous multimodal* functionality (i.e. create-once/publish many): a single source of textual or audio content that can be published as stand-alone editions, such as the way the Wall Street Journal uses the same articles for both its print and web editions. However, given that web consumers are already accustomed to a multimedia experience through their computers, the logical avenue of exploration is whether the appropriate mobile interface is similarly media-rich, or more specifically, whether VoiceXML is up to the challenge of *synchronous multimodality*. To be fair, no claims of multimodal support were expressed or implied in the 1.0 specification itself, but the topic

potentially speaks to VoiceXML's long term viability: if consumers demand visually and aurally rich mobile applications, then what is the role of VoiceXML in that multimedia environment?

13.6.1. VoiceXML + WML

Many of these same questions were being put to the gatekeepers of HTML: Can the HyperText Markup Language be extended to support a complex speech application? Before the clear dominance of VoiceXML at the W3C, there were indeed competing visions of how to create a voice-centric service language. Extending HTML had two advantages: leverage existing HTML software and hardware infrastructure, and the innate ability to support multimedia applications.[178] The first advantage is co-opted by VoiceXML's foundation in XML, now widely considered to be the more appropriate foundation for all new publishing languages rather than HTML directly. As for the multimedia advantage, its XML pedigree would suggest that the most obvious solution is to pair VoiceXML with WML to deliver a single application over WAP-enabled phones.

In anticipation of this marriage between WML and VoiceXML, the W3C and the WAP Forum held a joint workshop on the Multimodal Web in September 2000.[179] Even a cursory view of the attendees' position papers demonstrates a multiplicity of suggestions for delivering multimedia applications. Some advocate extending WML and VoiceXML to synchronize links to each other, others suggested an intermediary and non-interface depending markup language that could be transcoded into WML and VoiceXML as needed, while others called for scrapping VoiceXML/WML entirely in favor of creating a new multimodal language. The result of this conference was the *Hong Kong Manifesto*[180], which outlined the need to create a new working group to synthesize the concerns of the attendees and to create actual use scenarios to drive the next phase of the standards process. Within the W3C, the task has been taken up by the Multimodal Systems subgroup of the Voice Browser Activity, which has gone as far as releasing a multimodal requirements list for voice markup languages.[181] In some respect, there is no rush to find a solution given that the current generation of WAP phones do not allow concurrent use of the browser and telephony functions-examples that combine the use of HTML and VoiceXML were given but of limited interest to the mobile community. The drive to create a new standard in whatever form will be sustained

[178] Recall from the Technology Chapter that the most notable of these *HTML extension* efforts continues to be Microsoft's Web Telephony Engine and its five extensions to standard HTML. See http://msdn.microsoft.com/library/psdk/webte/wtestartpage_61et.htm.
[179] See http://www.w3.org/2000/09/Papers/Agenda.html.
[180] See http://www.w3.org/2000/09/Papers/Meeting/recommendations.html
[181] See http://www.w3.org/TR/multimodal-reqs.

by the combined growth of the WML and VoiceXML communities, neither of which has materialized to the extent anticipated.

13.6.2. Transcoding

Another approach to the creation of a voice user interface alongside existing HTML and WML products is to permit a real-time tool to parse the visual markup page directly and convert it into a speech-interactive dialog. This approach goes one step beyond simple screen-scraping, since the entire business logic and dialog flow is inferred from the visual page. In other words, the target XML for one technology is converted or transcoded into a page suitable for telephony interaction. The promise of this technology is compelling, since it implies that an application provider need not be concerned with the details of implementing a speech-interactive service because it will arise automatically from the existing HTML site.

Perhaps the most compelling and successful example of transcoding is the conversion of existing HTML pages into Compact-HTML[182], a subset of HTML created by ACCESS Co and adopted by NTT DoCoMo's i-mode[183] service as the standard delivery language. While programmers could certainly author C-HTML code directly, there are a variety of tools to perform these conversions automatically. However, the conversion process is further facilitated by the fact that both HTML and C-HTML are visually-oriented markup languages. While the programmer's task of negotiating the cellphone's smaller screen is a challenge, the same content flow and structure can be largely retained; in contrast, the dissimilarity between visual and aural interfaces presents another dimension of complexity that is difficult to overcome: human perception. While a person can parse and act on parallel sources of visual information, humans are capable of focusing on a single, linear stream of aural information. One simple example of this discontinuity is the hierarchical, menu-driven structure of HTML sites versus the linear, keyword-driven interfaces of well-designed speech sites. For example, selecting 1 element from a drop-down menu of dozens is standard web browsing fare, whereas enumerating the choices singly would unavoidably frustrate the user. In addition, transcoding technologies rely extensively on TTS technology to render textual information over the phone's speaker, resulting in an experience that compares unfavorably with the carefully-considered sites of voice-centric applications.

---

[182] See ACCESS Co's submission to the W3C, Compact HTML for Small Information Appliances, at http://www.w3.org/TR/1998/NOTE-compactHTML-19980209/.
[183] See http://www.nttdocomo.com/i/.

13.6.3. Short-Term Reality

Despite the momentum of the XML movement within the electronic publishing industry, the dominance of HTML focuses many resources on simply deploying a superior site without regard for possibility of publishing to other media formats. As a result, the centralizing force within a firm's information infrastructure is the data itself: text files, audio streams, etc. If the web architecture unit is charged with implementing a new access point to the data core, it will just as likely create custom WML or VoiceXML directly from the data than take the effort to unify all of its publishing efforts under a single XML framework. This does not imply that VoiceXML will not be dynamically-generated-to the contrary, it will likely become the most common mode of markup generation for nontrivial destinations-but it does mean that the site will be created from the data itself. While WML coding is very similar to HTML coding within the confines of a smaller screen, programmers are generally unfamiliar with the appropriate dialog structures and events that form the backbone of speech interaction, so the expectation is that VoiceXML code will be tightly monitored and hand-crafted in the short-term. As the publishing community continues to develop its familiarity with both XML structures and voice applications, there will be a shift to the multimodal programming described above. One must appreciate that a voice application provider is first and foremost concerned with delivering a compelling speech-interactive experience, and would be unwilling to compromise that position for the mantra of multimodality.

13.7. Utility: Multi-Document Grammars

13.7.1. Motivation for Interpreter-Side Grammar Management

The foregoing discussion has focused on utility issues found within a single VoiceXML document, but no mention has been made about the larger structure of the voice applications themselves. The creators of VoiceXML were keenly aware of the unique challenges of creating a language that could support the real-time input demands of a speech application with a client-server schema, and they did not waver in breaking with XML conventions when the need arose. For example, VoiceXML is endowed with many logic flow tags (e.g. if/then, goto) not found in other XML schemas, suggesting that VoiceXML is more of a procedural than a true markup (declarative) language. This compromise appears justified, however,

given that the programmer would prefer to avoid burdening the business logic apparatus with every single utterance from the user.Intranet traffic between the interpreter and the document server should ideally be limited to returning valid inputs and creating the next-state dialog. In this situation, another requirement is the ability to reprompt after timeout or failure to match against the reference grammar, a fairly common occurrence for a voice application.

Achallenge that occurs frequently in speech applications is the need to perform *fast context-switching*. A *context* is the environment within which a speech fragment must be considered, which in practice means the sum total of the active grammars at the specified point within the application flow. The context for any query in isolation is simply the grammar supplied to the ASR engine, but more complicated and useful contexts can be constructed as multiple queries are strung together. For example, suppose a portal application provides 'news' and 'sports' as top level elements, each of which grows into an ever-widening and -deepening tree of relevant audio content. If the user is 5 levels "down" into the 'news' and would prefer to listen to sports, she would certainly prefer to cut straight over to sports than to work her way back up the news tree to the main menu. In this example, the context is not only the operative grammar at her level within the news tree, but also the grammar active at the main menu.

The use of multiple active grammars as a single context is not limited to facilitating shortcuts through dense content trees. For example, a *mixed initiative* query permits more flexible input of data from the caller. Instead of leading the caller through a canned sequence of questions typical of a *machine-directed* query, the grammars for each of the individual questions can be simultaneously active, with the result that the user will be prompted only for missing information, if any. In this case, the meta-context is the union of individual questions, with control passing to each individual context after each utterance. Because of the increase of LAN traffic and the management overhead at the document server, context switching, (whether in support of hierarchical navigation or mixed initiative queries) is implemented on the interpreter using *scoped grammars*.

13.7.2. Application Root Document

As discussed in the Technology Chapter, the scope of a grammar defines its sphere of influence. Scoping is commonly encountered in computer programming: variable precedence and visibility is defined by its scope within the application. For example, the index variable of a FOR LOOP has a scope limited to the loop fragment, but variables defined just outside the loop are still visible to the expressions within the

loop. Grammar activity and precedence is generally determined by proximity, so a grammar with field scope has precedence over a grammar with document scope if there is a conflict, but both grammars can be active simultaneously. The highest level of scope (and therefore the lowest precedence level) is the document scope within the *application root document*. So long as each VoiceXML document within an application points to this root document, all of the elements defined within it-not only grammars but also scripts, links, event handlers, variables-are visible to the entire application. More specifically, the application root document is the only place where grammars can be visible to the entire application. **Whereas there are multiple levels of scoping within a single VoiceXML document, there is a single grammar scope that can be visible across multiple documents.**

Since a global command is nothing more than a context switch to the highest level of the application, the application root document is to the key to implementing universal functionalities. Through judicious application of grammars at various scoping levels, the programmer can support fast context-switching to any level of the program hierarchy without the need for record-keeping at the document server nor explicitly enumerating the global commands within each flat grammar.[184] The responsibility for defining grammars ultimately rests with the programmer, but now the management of activating/deactivating grammars and switching program control is handled by the VoiceXML interpreter itself.

Audible.com Example #1. Consider the following application: Audible.com's traditional business is to facilitate the download of various types of audio programs: audiobooks, lectures, public radio programs, newspapers, etc. Subscribers can either play the program through their computer or through a portable MP3 player. Suppose that the programming team has been charged with creating a voice application prototype, allowing access to only a subset of the available content. The application designers determine that they want users to be able to jump around content categories quickly, so the grammar for each dialog must not only account for selecting an audio file, but also switching between categories and global commands. The team quickly comes up with the following design:

- **Menu**. The user will enter the application through a main menu, offering a selection of 3 categories: business, spirituality, or mystery.
- **Content**. When a category is selected, the caller will be presented with the two most popular books in that category. By selecting the name of the book, the system will play a short audio clip from the book.

[184] The programmer must recognize that as the number of active grammars grow, so to does the complexity of the matching process within the ASR, potentially jeopardizing performance and accuracy, so the general rule is to keep the scope and number of active grammars as narrow as possible for the given functionality.

- **Global Navigation**. At any time, the caller can say a category name to switch over directly without having to first go through the main menu. In addition, the words 'main' or 'home' will take the user back to the main menu.
- **Other Global Commands**. 'help' is reserved for a short audio overview of how to navigate the system and speech recognition in general. 'goodbye' and 'exit' are both interpreted to mean that the user is done using Audible's voice application, so the system should just hangup.

Given the criteria above, the following is a highly stylized view of the VoiceXML documents required to implement this program:[185]

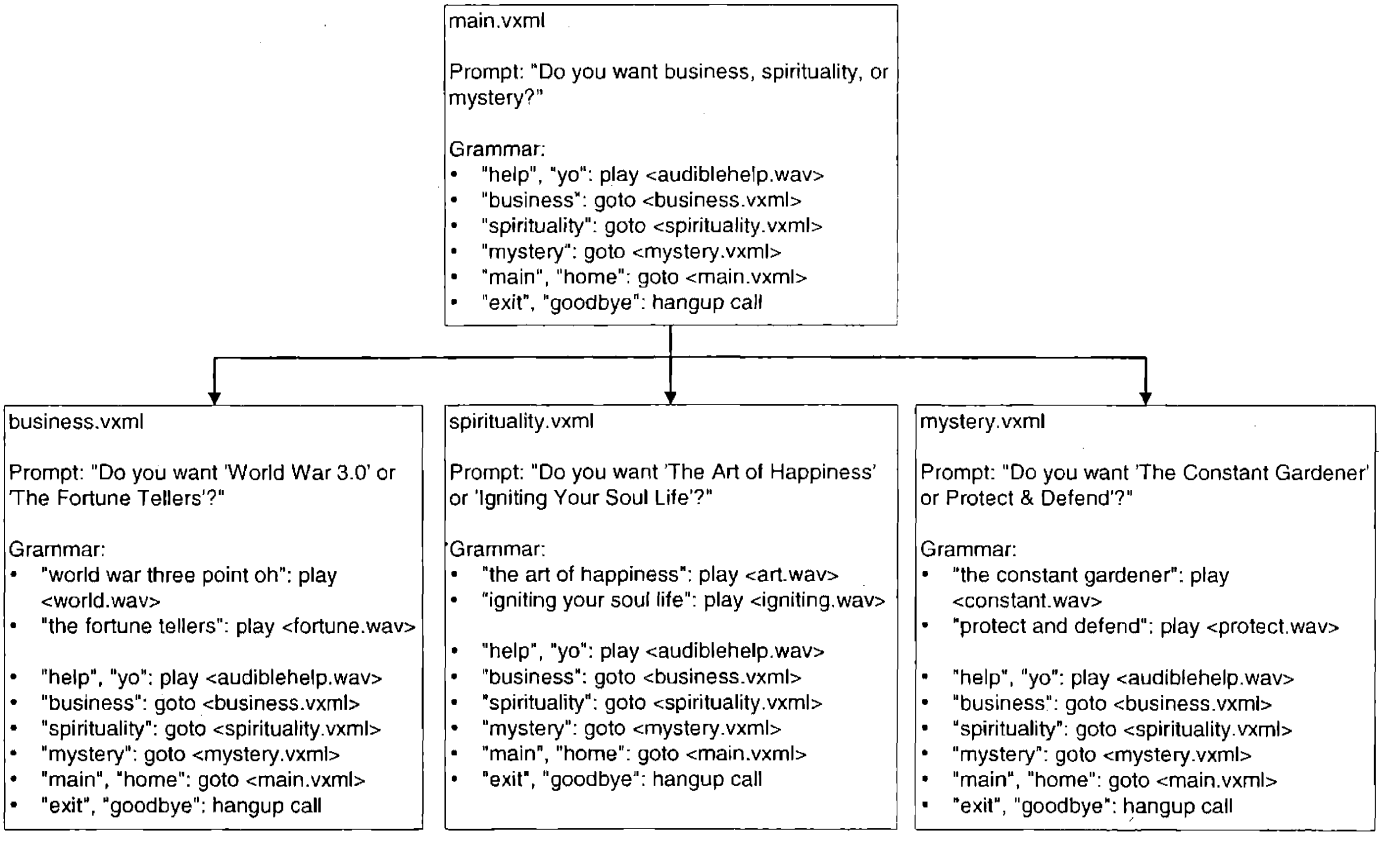

Figure 52. Callflow & Content for Example #1

The programming team recognizes that the global categories and commands appear in every document, so to both promote economy (keep VoiceXML documents short) and ease of editing (a single change in the

[185] The author realizes that a real application would provide much more permissive grammars, helpful prompts, etc. This example is meant only to illustrate the utility of multi-document grammars, not the elements of good voice application design.

global commands should not require editing 4 separate files), the programming team avails itself of the multi-document visibility of grammars that are defined in the application root document. The only changes required to achieve this effect is to designate main.vxml as the application root document and eliminate the global grammars from the 3 category-level documents:

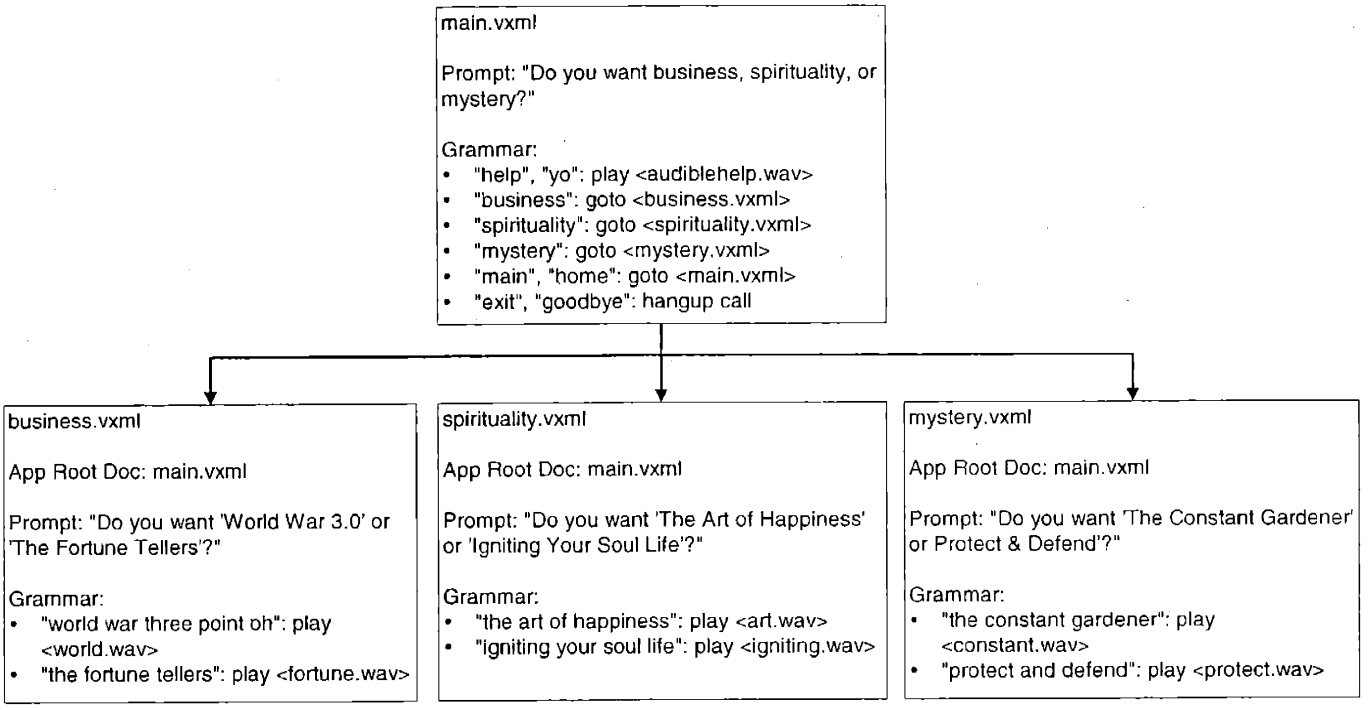

Figure 53. More Compact Representation of Example #1

This version of the VoiceXML and grammar code is **functionally identical** to the previous version: the caller retains access to all the global commands because the VoiceXML interpreter is instructed to include the grammars in the main.vxml document as well. **In other words, the scope of the global grammar is active across all the VoiceXML documents that refer to the application root document**, hence the name *multi-document grammar*. Since this type of navigation occurs frequently in speech applications, the designers of VoiceXML have met the challenge of supporting grammar management for this particular example.

Audible.com Example #2. Suppose that the Audible.com management team is suitably impressed with the prototype application, and thereby reward the programmer's efforts with the green light to extend the application to Audible's entire repository of audio content. The team drafts the following updated design rules:

- **Menu**. The user will enter the application through a main menu, offering a selection of all 22 categories.
- **Content**. When a category is selected, the caller will be presented with either of the following:
    - Another menu that enumerates the subcategories within the current category. Selecting the name of the category will lead to the next level below.
    - A list of audio titles available in that category. Selecting the title plays the short audio clip.
- **Local Navigation**.
    - At any given point in the category navigation, 'up' and 'back' are reserved for bringing the user back up the category tree one level.
    - At any given level in the category navigation, the caller can say the name of any parallel subcategory. For example, suppose that the user was given the choice of either 'Investing' or 'Leadership' while surfing within the 'Business' top-level category. If the user chose 'Investing' but then decided instead to listen to the options within 'Leadership', the caller would not have to say 'back' first and then the subcategory name.
- **Global Navigation**. At any time, the caller can say a **top-level category name** to switch over directly to the top of that category without having to first go through the main menu. In addition, the words 'main' or 'home' will take the user back to the main menu.
- **Other Global Commands**. 'help' is reserved for a short audio overview of how to navigate the system and speech recognition in general. 'goodbye' and 'exit' are both interpreted to mean that the user is done using Audible's voice application, so the system should just hangup.

So other than expanding the breadth and depth of choices, the only changes to the user interface are the addition of 'back' and 'up,' and the ability to jump to parallel categories within the category tree. Using the "full" version of the code without multi-document grammars for the purposes of illustration:

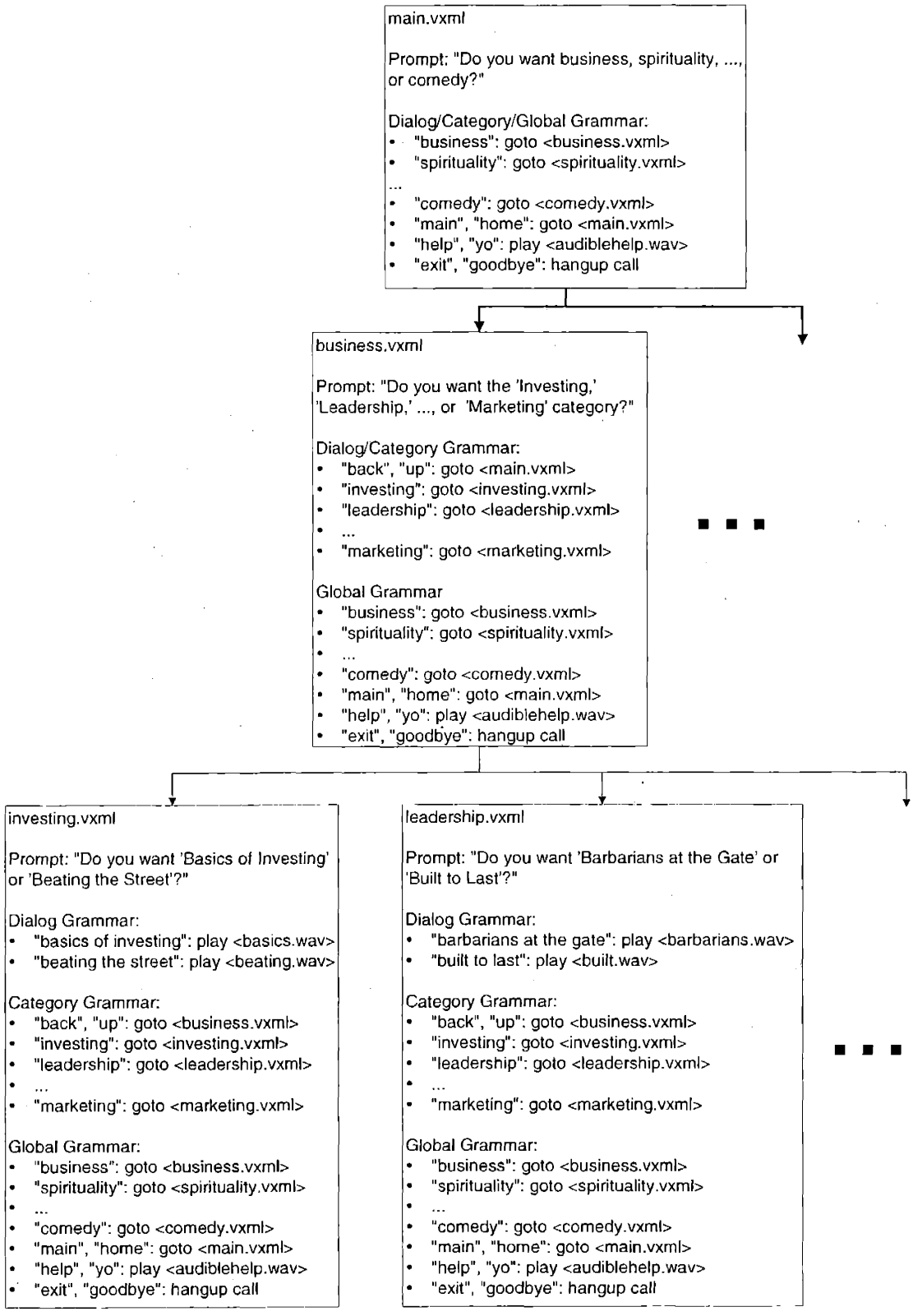

Figure 54. Callflow & Content for Example #2

In order to simplify the grammar management process, the programming team would once again prefer to avail itself of the multi-document grammar functionality afforded by VoiceXML. Specifically:

- The *Global Grammar* should be put into main.vxml so that it is visible to every document "under" it.
- The *Business Category Grammar* should be put into business.vxml so that it is visible to every document "under" it, such as investing.vxml and leadership.vxml.
- The *Dialog Grammars* simply remain inside their respective documents.

However, the reader will recall the earlier caveat that there is only **one** grammar scope that is visible across multiple documents for the entire application. Specifically, the *application root document* pointer in each page can only point to one file. Once the VoiceXML interpreter encounters a document that points to a different root document, all the knowledge from the previous root is entirely discarded. In fact, the VoiceXML specification defines an application as "a set of [VoiceXML] documents sharing the same application root document."[186] The development team has a problem:

- If only the Global Grammar is put at (?) the application root, then there is no possibility for making the Business Category Grammar visible across the multiple documents within that category expect for exhaustively enumerating them as above. This problem is not just an isolated instance, since there are dozens of Category-level Grammars as the user traverses both down and across categories.
- If only the Category-level grammars (e.g. 'Business') are put into the application root, potential conflicts arise between grammars that share names:
    - Both the 'History' and 'Science' subcategories each have a section titled 'Biographies.'
    - In fact, the 'History' category has its own section named 'Science'
    - 'Up' and 'Back' clearly cannot be included in a single global grammar since their usage context defines the appropriate action
- Folding both grammars into the root grammar does nothing to eliminate the potential conflicts already mentioned, but rather it exacerbates the problem of incorporating this application into a single portal application (more will be said about this later.)

Given that VoiceXML provides no appropriate solution to this situation, the programming team has no choice but to exhaustively enumerate the grammars at every level, with the possible exception of the application's global grammars that should be visible to all pages. If the VoiceXML code is dynamically

[186] VoiceXML 1.0 Specification, Section 3.3.

generated, the development team may take the initiative to create its own grammar scoping constructs within the business logic itself, so that the process of enumerating every appropriate grammar possibility for every dialog is largely automated by the server code. There is the requirement that programmers handle grammar scoping through the interpreter, but it certainly violates the spirit of VoiceXML: XML programmers should not be burdened with the lower-level details of grammar management. VoiceXML is inadequate to the task of supporting manifold levels of multi-document scoping, a functionality that is clearly needed by voice application programmers, and this glaring omission is no more obvious than in the case of an audio portal.

Audible.com Example #3. With much rancor in the programming ranks, the team manages to pull together Java code to handle the annoyance of tracking Category- and Dialog-level grammars in order to dynamically resolve and generate the appropriate flat grammar for every page. The application root document, however, is preserved and is used to implement the global grammar... they might as well take advantage of what little multi-document grammar functionality VoiceXML affords. The management at Audible.com is duly impressed with the more comprehensive version of the application, and announces that a deal has been forged with the VoiceXML-based consumer portal Tellme.com: once callers connect to Tellme, the keyword 'Audible' will launch the Audible.com application. Supposing that all issues of application distribution and audio formats and grammar formats have been resolved satisfactorily, the following design goals are added to the application:

- It goes without saying that Audible.com's "global" commands are global only to its own application, and not to Tellme's main menu and parallel applications.
- 'Tellme menu' and '**' (star key pressed twice) are reserved for returning the caller to the main Tellme menu.
- 'Goodbye' returns the caller to Tellme and offers the ability to continue in the service before terminating the call with a closing note.

In order to enforce this uniformity across all of its application partners, Audible.com is instructed to reference the Tellme "universals" file (located at http://resources.tellme.com/lib/universals.vxml) as the application root document in every generated page, thus freeing the Audible programmers from having to concern themselves with actually implementing these new global grammars. In other words, simply referencing the Tellme universals as the root document will handle the global return to Tellme menu and graceful call termination functions.

The addition of the Tellme main menu "application" at a level above the Audible.com application creates problems completely analogous to those produced when the Audible application itself was expanded to multiple levels. The programmers are faced with the same difficult choices: The Tellme grammar has to be visible at all times within the Audible application, and Tellme reserves the right to change this grammar at any time, so the Audible programmers must comply by changing every application root pointer to this Tellme file. However, this means that the formerly global grammar is now only global to the application itself, so it suffers the same fate as the rest of the Category-scope grammars: it must be exhaustively enumerated for every VoiceXML document. In other words, the Audible global grammar is now just a grammar for the "category Audible."

However, the inclusion of the Audible application gives rise to a contention not encountered earlier. The Audible application reserves the word 'goodbye' as the termination of the call, whereas the Tellme universals grammar offers a more verbose option of confirming the exit or continuing with another Tellme offering. Since the single level of multi-document grammars forced the programmers' hands into putting all the Audible commands into a single grammar, 'goodbye' is defined as a grammar of document scope; in contrast, Tellme's grammar reference for 'goodbye' occurs in the application scope. Since scoping precedence is determined by proximity, Audible's definition of 'goodbye' will win every time, which is exactly the opposite behavior desired by the portal operator. Without any malice on Audible's part, it has inadvertently wrested control from Tellme. The implication for all voice portals is humbling: the portal's global commands can be co-opted at any time by the external application.

In the case of a portal that creates all of its own applications, this conflict can be resolved by committee: all applications must conform to a design standard such that these conflicts simply do not arise. To a lesser extent, this design logic can be extended to external applications: "If you want to be part of the TellMe consumer portal, your application must conform to the following principles and refrain from using the following verboten phrases...." This closely reflects the AOL model of including external content: HTML pages must conform the to its specifications (e.g. graphics layout, only subset of HTML supported) before they assigned a keyword and become an integral part of the AOL experience: web surfing is clearly a discouraged behavior on the AOL network.

This case of Audible residing within another audio portal demonstrates that the existing support for multi-level grammars not only forces programmers to handle the entire grammar management process on the server-side, but the engineering precedence order directly conflicts with the business logic of the portal itself: the portal's commands must always supercede those of any single member application. In the case

of Tellme, the number of global commands is quite small, so it would not be too prohibitive for the Audible team to create a version of the code specific to the Tellme portal. As the number of portals grow, this process of modifying the Audible application to satisfy the whims of each and every portal is unscalable and creates a code management nightmare, and it clearly cannot support a true Voice Web architecture.

13.7.3. Potential W3C Grammar Scoping Contribution

Just as W3C's DialogML contribution may address the audio issues mentioned earlier, it may also address some of the limitations for scoping. In the 'Variables' section labeled as "must have":

> "Variables can be scoped within namespaces: for example, state-level, dialog-level, document-level, application-level or session-level. The markup language defines the precise scope of all variables... The precise requirements on variables may be affected by W3C work on modularity and XML schema datatypes. [187]

The W3C is apparently considering the introduction of a level of scoping above application-visibility: session-level, which is presumably active for as long as the caller is being serviced by a single instance of a DialogML interpreter. Applying this to the Audible/Tellme example above:

- Tellme's truly global commands will be part of the session grammar
- Audible's application-global commands will be part of the application grammar

However, if this is the extent of the solution, then DialogML's grammar scoping will still be limiting to programmers. First, hard-coding a single "extra" level of scoping above the application level fails to address Audible's own problem for "category-level" scoping.[188] Multi-document scoping is still an all-or-nothing proposition for the voice application programmer, so only Tellme potentially benefits from this additional scoping layer.

And even that benefit is limited: There is no explicit mention that the normal precedence rules for scoping will be abrogated for the session-level grammar, so Tellme's global commands are still held hostage to

[187] Section 3.5 of Dialog Requirements for Voice Markup Languages, W3C Working Draft, 23 December 1999. See http://www.w3.org/TR/voice-dialog-reqs/.

any single application, e.g. until Audible explicitly removes it, Audible's 'goodbye' still trumps Tellme's. If one further supposes that Tellme desires to make its consumer portal available to other portals, does the target portal then co-opt the session-layer grammar, forcing Tellme to steal the application-layer grammar from Audible and all the other Tellme applications?

The session-layer grammar is a point solution that doesn't address the problem at the core of multi-layered applications (whether they are standalone apps or whole networks of applications.) Given the prevalence of consumer portals that take responsibility for the production of the meta-application and the constituent applications as well, this problem may be dismissed as relatively unimportant. But as applications grow in depth, either through growing a single application or by building a network of applications, this issue of flexible multi-document scoping cannot be ignored.

13.8. Voice Web-Specific Standards Issues

The problem of scoping is no more evident than in the Voice Web model: not only is the entire panoply of issues inherited from the portal example above, but there is not even the opportunity to resolve these conflicts through committee *a priori*. For example, if Audible is serious about distributing its content through Tellme, it will willingly suffer through the necessary "impositions": generate all grammars within document scope, remove commands that conflict with Tellme universals, etc.-AOL is the proof-case that the prospect of reaching millions of consumers will stimulate firms to go to great lengths to accommodate the idiosyncrasies of a non-Web portal.

However, the Voice Web model does not afford this flexibility since any site is potentially reachable or linkable from within any other site, particularly one which would prefer to hold onto the caller for as long as possible. Audible may be willing to extend its resources to accommodate Tellme, but what about the potentially dozens of other destination portals: e.g. if the resulting industry structure is dominated by the cellular carriers themselves, does Audible have to undertake a multitude of custom programming tasks? Audible may be one of the fortunate few for whom application development costs would be subsidized by the portal itself (though this is an unsustainable proposition for the portal industry), but other content players may not be so fortunate.

---

[188] The reader should remember that there is no such programming construct as a "category-level grammar" per se. The author is referring to Audible's specific problem of desiring commands to be visible across **some** of the documents but not **all** of the documents, such as the quick navigation commands at the category level.

To be fair to the creators of VoiceXML 1.0, there is no claim expressed or implied that the language was well-suited for the challenges of creating a Voice Web network. However, this is indeed the commitment of the World Wide Web Consortium: "The W3C Voice Browser working group aims to develop specifications to enable access to the Web using spoken interaction."[189] So this section is dedicated to exploring those Application-Speech Gateway Standards issues that unique affect the formation of a Voice Web industry.

13.8.1. The Portal is not a Browser

If for the sake of discussion DialogML successfully addresses the scoping limitations of VoiceXML, then the remaining problem is reconciling the normal scoping precedence order with the need for universal grammars, especially those from other random organizations. Part of the solution to this conflict is the recognition that there is a difference between a *portal* and a *browser*, a distinction that is obscured by the linear interface of voice applications.

- A *portal* defines a business model of providing links to external content (though internal content is certainly permissible)-the portal and the target sites are implanted in XML and are subject to the standard rules of HTML interpretation. Examples include: Yahoo, Excite, etc.
- A *browser* is an interface that allows a user to direct the operation of the HTML interpreter. For example, the user can reload a page, stop a download in progress, view the previous HTML page, etc. This browser is not an HTML page itself, so it is not subject to the rules of HTML interpretation-it is an integral part of the browser. Examples include Internet Explorer, Netscape Navigator, etc.

Their obvious dissimilarity in the HTML context is masked in the VoiceXML framework by the one-dimensionality of the interface. An HTML browser has the benefit of visual layout to present and distinguish global and local elements of the surfing experience. Consider the following screen shot of one page within the Insurance Information Institute's website.

[189] Abstract of Dialog Requirements for Voice Markup Languages, W3C Working Draft, 23 December 1999. See http://www.w3.org/TR/voice-dialog-reqs/.

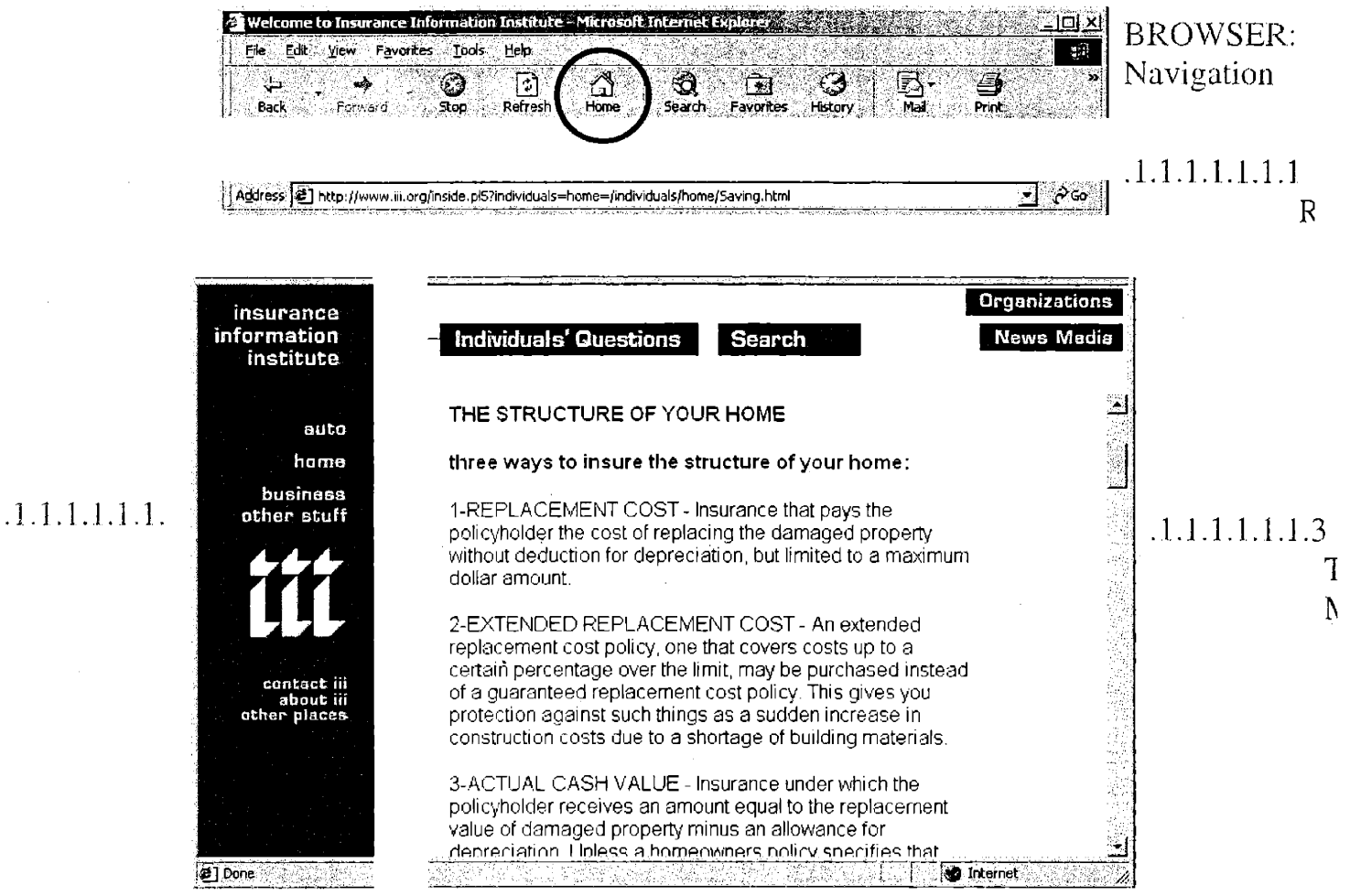

Figure 55. Universal & Local Elements of an HTML Browser

- **Browser**
  - **Navigation**. These commands are not specific to the site being viewed, e.g. the 'Stop' button will always prevent further processing of the incoming page, the 'Back' button will display the last HTML page (whether this page was on CNN's (don't you mean the Insurance Information Institute here? Where does CNN come in?) site or not), etc. Note that the destination site, the Insurance Information Institute, has no role in either defining or implementing the behavior of these commands.
  - **Uniform Resource Identifier**. Known more commonly as the *address* or *domain name* of the destination site, this is a string of ASCII characters that uniquely identifies this particular site and page anywhere on the HTML Web. While the logical or IP address of the site can change, the URI performs the redirection to maintain site availability. It also allows the web surfer to use a more mnemonic trademark or phrase than rely on a

numeric IP address. At any point within the clickstream, the user can enter a new address and be taken to the new site. Apart from registering the domain name, the Institute is not responsible for nor can it interfere with the operation of this URI service.

- **HTML Window**. Whatever appears in this window is the direct result of the interpretation of the HTML code supplied by the destination site. In this case, the site programmers have determined that there is a lot of information to present on their site, so they have opted to segment the HTML window into two parts: a menu that allows quick navigation to other top-level topics within the site, and a content window that contains text and links to other items.
    - **Menu**. This is a top-level menu, but only within the Institute's site, since it would disappear if the surfer changed the address to CNN.com, for example. This menu allows the user to change the topic focus entirely from home insurance, e.g. auto, business, etc. This menu will always be available regardless of where the user is within the Institute's site, and the Insurance Information Institute takes full responsibility for defining and implementing this menu. This is redolent of the fast category switch discussed earlier.
    - **Content**. This is representative sample of the Institute's actual product-textual articles of interest to the prospective insurance consumer. This information is the teleological goal of surfing, and all the other navigation elements, whether universally or locally defined, serve to facilitate the user's search for content.

Functionally speaking, the Browser elements combine aspects of a global grammar (visible in all context) with the precedence of a local grammar (overrides any potential 'conflict' between itself and anything encountered in the HTML window.) Of course, from a visual perspective, most web users are not confused by the appearance of the word 'Home' in both the HTML Window and the Browser Navigation bar: the former implies a movement back to the Institute's homepage, whereas the latter connotes a return to some default page that the user had defined earlier. But even if both options were simultaneously activated, perhaps by someone who's quick with the mouse, the Browser command would override any operation in the HTML window.

This simple, universal, highest-precedent element called the browser should not be viewed as the top level of a portal. First, the rules of grammar precedence would have to violated for the special case of the portal's global grammars. Another requirement is that these commands can interrupt the VoiceXML browser at all times, even if to stop the download of a page. For example, one cannot implement the 'Stop' function in VoiceXML, since it has to be available before the browser is ready to present the page; in other words, the browser is performing real-time functions that do not fit within the client-server

paradigm. Perhaps most importantly, no single site should be responsible for implementing the basic navigation features of the browser. The content and application communities are interested in phonecasting their consumer wares, not getting bogged down in the details of good browser design.

Potential W3C Browser Contribution. Once again, the W3C's DialogML requirements document hints at including this Browser-level functionality under the title of Meta-Commands (prioritized as "should have"):

> "The markup language specifies a set of meta-command functions which are available in all dialog states; for example, *repeat*, *cancel*, *quit*, *operator*, etc. The precise set of meta-commands will be coordinated with the Telephony Speech Standards Committee. The markup language should specify how the scope of meta-commands like *cancel* is resolved."[190]

The spirit is clearly approaching that of the Browser-level commands: highest-level of precedence such that programmers would avoid using them in their own code, they would be active at all times, with the implication being that the voice application programmer would not bear the responsibility of implementing these functionalities. If this 'meta-command' construct is functionally equivalent to the browser-level functionality described earlier, then the task remaining is to rally the industry around a standard single set of commands, a task that has apparently been relegated to the *Telephony Speech Standards Committee*.[191] This committee would also ostensibly be charged with defining the default actions for these commands, e.g. what should be played to the user when she says 'cancel'?[192]

While there may be widespread agreement that elements such as 'repeat' and 'cancel' should be part of the browser's *lingua franca*, it will be difficult to circumscribe the discussion in the face of pressures to add to the corpus. Even a cursory glance at HTML Browsers toolbars demonstrates this "creeping" functionality. For example, should *bookmarking* be the responsibility of the portal site or the browser

---

[190] Section 2.6 of Dialog Requirements for Voice Markup Languages, W3C Working Draft, 23 December 1999. See http://www.w3.org/TR/voice-dialog-reqs/.

[191] Despite the author's best efforts, there is no mention of this organization within the W3C or the VoiceXML Forum. The closest reference that the author can uncover is the *Telephone Speech Standards Committee* whose website has not been updated in almost 2 years: http://www.delphi.com/speechreco. Some older information can be found at: http://www.speechcentral.com/content/tssc.html. The author sincerely hopes that such an organization is indeed actively engaging this issue for the Working Draft release of DialogML.

[192] Interestingly, there is no parallel user interface standard for HTML browsers, but most of these standards conform to the W3C's User Agent Accessibility Guidelines at http://www.w3.org/TR/UAAG10/. Browsers, however, have been following these conventions long before the Guidelines were finalized, though they may have sometimes gone by different names, e.g. Internet Explorer 'Refresh' and Navigator 'Reload' are functionally equivalent.

itself? Section 3.9 of the requirements list suggests the inclusion of streaming audio types, which would necessitate the addition of long-play audio navigation to the browser's argot. Other features could include: user profile and preference management, e-commerce wallet, user-defined hotwords, context-sensitive tutorials. Particularly because of callers' intolerance to changes in the telephony user interface, there is a strong argument to be made for requiring these types of functionality in all VoiceXML browser implementations. Proprietary, platform-specific solutions such as Nuance's *Voyager*[193] product correctly anticipate the need for this rich browser layer above the core feature of VoiceXML interpretation, but content and application providers will prove resistant to adopting any technologies that destroy the platform portability that VoiceXML ostensibly promotes. The W3C should seize this opportunity to standardize the most commonly requested browser features before they become co-opted by proprietary and irreconcilable implementations.

Practical Limits to Scoping (a.k.a. Nesting is Not Normative.) The integration of the speech-driven browser interface is an important tool for supporting the precedence and universality of the most common commands, but this still does not address the more general need for precedence inversion. For example, in the Audible/Tellme example above, Tellme's commands are not part of the browser's universals any more than Yahoo's top-level menu is integrated into Internet Explorer: there still needs to be a mechanism by which a higher-level grammar (e.g. portal grammar) can supercede that of a lower-level grammar (e.g. individual portal application.) Given the reasonable requirement to support the traditional conventions of precedence in scoping levels, there may be no solution other than brute force customization for each portal.

However, one must step back from this obsession with supporting grammar scoping across organizational boundaries and postulate the most common mode of access. The HTML web suggests that portals, such as Yahoo and Excite, do not generally hold onto to users by remaining active throughout the web surfing session. While these types of organizations do indeed attempt to keep the user as long as possible by continually adding and reselling services under the portal brand, their main function is to direct and release the consumer. In this sense, the HTML Web is a very *flat* or *serial* experience: there may be great depth within a single organization's code, but sites are generally not made to work as sub-applications within other sites. Consider the following worst-case scenario of requesting stock information through Ask.com:

---

[193] See http://www.nuance.com/products/voyager.html.

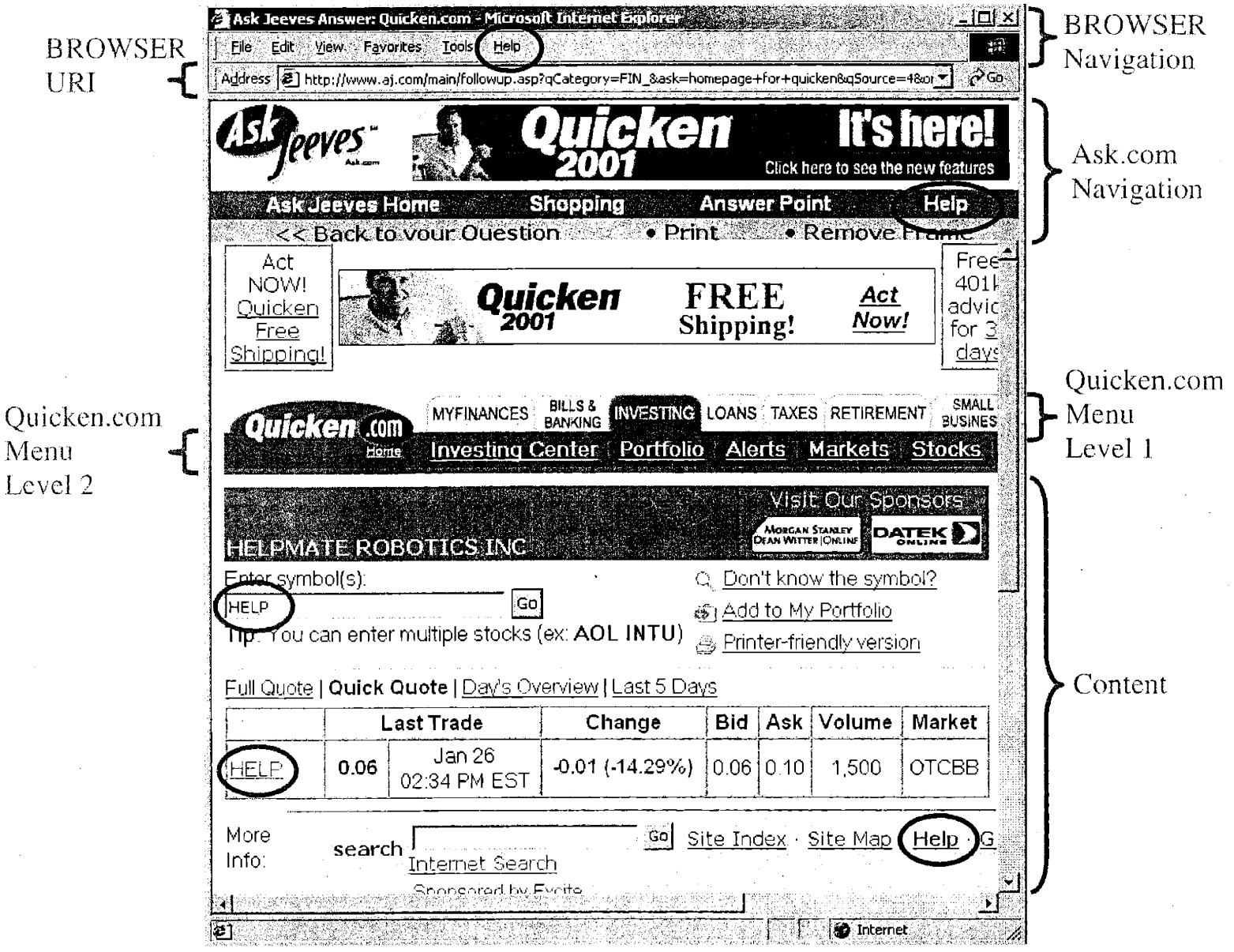

Figure 56. Various Scoping Levels within an HTML Browser

The Browser, Ask.com, and multiple Quicken 'commands' are available at the same time, and their layout on the screen (i.e. context) differentiates their meaning and mode of activation. However, one would not make the claim that Quicken.com is an application within the portal Ask.com. Ask.com has certainly opened a new frame such that the Quicken.com site can operate independently, but there are no hierarchical or functional relationships among these various elements within the HTML window. For example, the word 'Help' appears 5 times in the diagram above and each has a different meaning, but

browser is not responsible for resolving any contention between these uses since each use is defined independently.

Apart from the fact that such a surfing experience would be too difficult to conceptualize for a user limited to an aural interface, this is not a representative use of the HTMLWeb. There will certainly be an enduring role for audio portals and other aggregators, but the Voice Web model encourages the development of parallel, peer sites that are accessible through a minimal set of the browser (not portal) commands. Audible.com may indeed profit much from a keyword alliance with Tellme and other major portal players, particularly in the short-term, but phonecasters are looking for a more meaningful and permanent solution that parallels the independence and autonomy fostered by the HTML Web.

13.8.2. Vocal Domain Name System

An indispensable aspect of every HTML browser interface is the ability to enter a (hopefully) mnemonic sequence of characters in order to access any site on the HTML Web. The presence of this feature is the democratizing force of the Web, since site-to-site excursions are not dependent on the HTML contents of the default site. In the case of the HTML Web, this sequence of characters is most commonly known as the web address, but is more formally known as a *uniform resource identifier (URI.)* While the HTML address is only one of many possible URI schemes (e.g. 'mailto'), it is by far the most common.[194] Once a domain name has been accepted by accredited Registrars, it is propagated to the authoritative and regional Domain Name Servers appropriate for that top-level domain. This global network of DNS servers is responsible for resolving each host name into a numeric IP address, e.g. hotwired.com resolves to 204.62.131.129. The routing element of the Internet understands only these numeric IP addresses, so domain name resolution must be successfully completed before the first(?) packet is retrieved from the destination site. The browser does not require the use of domain names, but it would be very cumbersome to advertise and remember arbitrary numeric sequences for branded sites.

There is a uniform IP addressing space, but there are a variety of services that operate on top of the IP layer, e.g. http. The URI convention was specifically designed to accommodate new services that rely on IP: each new protocol or document format need not require its own DNS service. Some services have their own URI schemes (e.g. 'mailto') while others simply borrow the existing HTML convention (e.g. WAP addresses utilize the familiar 'http' scheme.) However divergent the actual services delivered

[194] See http://www.w3.org/Addressing/schemes for a partial listing of URI schemes.

through URIs and the DNS system, the tie that binds them together is the assumption of *text addressing*. For example, WML and HTML sites rely on entirely different document and protocol technologies to implement their respective services, but both rely on text addressing: e.g. http://wap.tdwaterhouse.com and http://www.tdwaterhouse.com. There is no denying that entering in long addresses through the WAP kaypad interface is cumbersome, but the resulting addresses is structured from the same subset of about 60 ASCII characters.[195]

Since the Voice Web, like any other network of peer sites, fundamentally relies on ad hoc and democratic access to any site connected to the Internet, there must be a corresponding mechanism of specifying IP addresses in an intuitive manner. There is no technological reason why callers can't simply speak each digit of the IP address for destination sites, but human factors considerations demand a DNS-like service for the Voice Web. Unlike WAP, however, the Voice Web system must account for challenges of speech-directed URL specification. The most obvious challenge is to match a single utterance against a library of potentially hundreds of thousands and to disambiguate the correct selection from other likely matches-there is a real concern of scalability over the long-term. Apart from the sheer scale of the name resolution system are the more mundane recognition concerns: difficulty in distinguishing short names with similar pronunciations (bee.com vs. pea.com), true homonyms (carrot.com vs. karot.com, I.com vs. eye.com), etc. From a regulatory perspective, the final solution must address the potential trademarking contention caused by homonyms and preserve the vendor independence of the DNS system. Given these technological, regulatory, and human factors constraints, there are several potential solutions.

- *Adjunct Speech-to-Domain Service*. Just as each ISP doesn't support its own DNS servers, nor should Voice Tone Providers be required to perform this resolution themselves. When the caller specifies a new URI, that sequence is captured as an audio fragment and sent to the new Speech-to-Domain Service (SDS), which is a network of servers whose **only** task is to recover the URI string from the utterance. SDS servers would always have the most current DNS global database and be optimized for matching against this large, compiled grammar. Such a process would invariably result in multiple matches, so the SDS would also initiate a disambiguation session with the user directly. Once the desired URI is identified, it is sent to the nearest appropriate DNS server for IP resolution. This configuration does not burden the VTPs with the customized task of URI extraction, nor does it require any changes to the existing DNS system. This arrangement in

[195] The official URI syntax is defined in RFC 2396. See ftp://ftp.isi.edu/in-notes/rfc2396.txt.

no way trivializes the process of identifying a single entry from the entire DNS universe, but it does allow for task specialization and supports minimal invasiveness.

- *Adjunct Speech-to-Domain Service with Flags.* This is the same architectural solution as above, with the addition of a speech recognition "assist" from the DNS system itself. The full DNS record for each domain name has room for much more than just the IPv4 address. There are a variety of address and miscellaneous fields already supported within the canonical DNS record, e.g. X.25 PSDN address, ISDN address, IPv6 address, security key and signature, etc.[196] From a technology standpoint, it would be a trivial matter to add a single Boolean entry called "Voice Site Present." A '1' value would indicate that the domain operator does indeed offer a voice site at that address, whereas a '0' value would imply that no such capability exists. The performance and accuracy boost from this single bit is significant since it reduces the search space from hundreds of thousands to potentially hundreds, especially in the formative stages of this industry. The SDS system would monitor the master DNS database and compile only those marked records into the domain grammar. A variation of this system that doesn't require modification to the DNS entries is to store that Boolean entry at the SDS system itself: voice site operators would report the existence of a Voice Web site to the SDS system directly instead of the traditional DNS authorities. Regardless of whether this flag is stored in the DNS or SDS records, the search space is dramatically reduced and otherwise functionally equivalent to the vanilla SDS system.

- *Adjunct Speech-to-Domain Service with Flags and Phonetic Assist.* As the name would suggest, this schema is functionally equivalent to the previous proposal with the following addition: along with setting Voice Site Present Flag, site operators will pass along the phonetic pronunciation(s) that match their domain. The inclusion of the phonetic "assist" further reduces the speech recognition complexity by essentially providing the name-to-grammar mapping that the SDS would otherwise have to infer. For example, faoschwartz.com is not pronounced as a single word but as 'F A O Schwartz'-rather than simply hoping that the SDS will have the knowledge to correctly map the phonetic pronunciation, the domain owner can explicitly provide the correct phoneticization. Given that the W3C's Speech Recognition Grammar and Speech Synthesis Markup Languages both anticipate support for the International Phonetic Alphabet (IPA), this

[196] The authoritative list of DNS entries is found at ftp://ftp.is.co.za/internet/in-notes/iana/assignments/dns-parameters.

would be the appropriate choice for declaring the correct pronunciation to the SDS system.[197] In effect, each domain operator would be responsible for populating his phonetic entries in the master domain grammar. In order to avoid the trademarking concerns of domain hijacking or squatting, the only permissible phoneticizations are those that follow the registered text domain. For example, if FAO Schwartz did not own faoschwartz.com but did own fao.com, it could not take this opportunity to expand the phonetic pronunciation to the entire name-it would have to resolve the dispute with the owner of the desired URL through the normal DNS contention processes. Again, this phonetic assist information can be populated in the DNS record directly or the adjunct SDS system.

- *Phonetic Alphabet is Just Another Language*. Perhaps the most *avant garde* approach is to treat the International Phonetic Alphabet as its own language, which would require that organizations compete for domain names in the standard DNS process. The W3C is finalizing its Internationalization process of amending the URI character conventions to utilize the more capable UTF-8 (8-bit Unicode) character set[198], which flexibly allows any known alphabet to be used in specifying a URI, e.g. http://[redacted].com. So not only would companies have to protect their trademarks in the English language domain system, but in all known languages, including the phonetic language-the normal processes of trademark litigation and first-come-first-served would apply. The benefit is that the SDS system would interpret the utterance within the context of the universe of phonetic domain names. In other words, the Voice Site Flag and the phoneticization would not be fields within the English-oriented DNS record, but the phoneticization itself would have its own separate DNS record.

- *Independent Speech-to-Domain Service*. Instead of relying on the existing DNS system, an entirely parallel system can be built to cater to the needs of the voice industry. For example, the domain names do not need to be encapsulated within the arcane 'http', 'www' and 'com/org/net' elements. There is the added advantage that the search space would be much smaller, especially in the formative stages of the industry. The HTML Web precursors of Netscape Internet Keywords and RealNames Keywords (on Internet Explorer) provide a similar AOL-like keyword service through the URL fields of their respective browsers.[199] Apart from the administrative and

[197] Section 6.5 of the Speech Grammar Markup Specification and Section 2.5 of the Speech Synthesis Markup Language Specification mention the use of the International Phonetic Alphabet, which is standardized by the International Phonetic Association (http://www2.arts.gla.ac.uk/IPA/ipa.html).
[198] See http://www.w3.org/International/.
[199] See http://www.realnames.com/ and http://keyword.netscape.com/.

legal overhead of creating a parallel agency, the W3C clearly prefers the extension of the already flexible DNS system to the creation of point solutions for new services when possible. Note that each consumer portal currently runs a proprietary form of this independent SDS system whenever it assigns a keyword to an application partner, which could easily evolve into the revenue-generating scheme that AOL employs: keywords are acquired through outright leasing or through significant advertising commitments.

In any of the proposed solutions above, the ability to bookmark a site is not just a convenience for the consumer but perhaps the only means of ensuring a scalable SDS system: how can the SDS system scale if each request requires potentially multiple minutes to resolve due to user participation in disambiguation. There are always new methods that can be applied to improve the recognition performance (e.g. 'speak-and-spell' tactic whereby the caller both pronounces and spells the name to eliminate a significant number of false positives), but the SDS service would undoubtedly prefer to minimize redundant searches for the same user. Bookmarking may simply be a convenience on HTML browsers, but it may very well be the saving grace of the Vocal DNS system, so the W3C Voice Browser Working Group (or the aforementioned Telephony Speech Standards Committee) should seriously consider including the bookmarking function within the browser interface.[200]

Failure to create a Vocal DNS system will solidify the dominance of the portals and perpetuate the AOL-model of vertical integration. Even in the case of WML technology that requires no DNS augmentation, the limitation of *entering* the URI through the keypad has all but assured rents to the carriers that lease top spots on their default WML page. The Vocal DNS system must not only offer the challenging functionality of resolving spoken fragments into IP addresses, but it must also be easy enough for the user to engage successfully and bookmark the site when found.

13.8.3. Personalization and Customization

The final topic to be addressed before moving to the next element in the market value chain is the ability to customize or personalize VoiceWeb applications from a single set of consumer data. For example, a caller's preferred cuisine is available to food-related VoiceXML sites that can customize the presentation

[200] This Browser Interface group would also be similarly charged for standardizing the universal command to enter 'URI Entry Mode', since the caller should not expect to say 'pizza' at any time and expect to be taken to http://www.pizza.com.

of the content around that cuisine, a consumer's billing and shipping information is automatically and securely conveyed to the VoiceXML e-commerce site to encourage impulse buying, etc.

The short answer is simply this: the market should not expect the Voice Web to overcome the fundamental technological or market limitations of the Visual Web. These features can be turned on their head and posed as questions: "Why can't HTML surfers currently purchase any item from an e-wallet?" or "Why doesn't this random food site understood that I like Chinese food automatically?" For example, there have been isolated efforts to build e-wallets-Brodia Personal Commerce Manager, Yahoo Wallet, Microsoft Passport-but they only work on sites specifically designed to utilize them (which is a small fraction of the total commerce sites on the Internet.) The same reasoning holds for Caller preferences: most consumers are more concerned that this information will be shared without their permission than they are about getting this sensitive data in the hands of marketers and random destinations on the HTML Web.

The Voice Web experience may indeed be much improved if such functionalities were available across all destination sites, but VoiceXML does not provide the magic bullet for doing so. The impetus for creating this network externality has to come from the industry itself, and so far that support has met with little success. The road of good intentions is paved with the corpses of standards that nobody wanted in the first place (e.g. W3C's Platform for Internet Content Selection[201]), a lesson that the W3C in particular seems to have difficulty in learning (e.g. W3C's Platform for Privacy Preferences[202].) In summary, these issues may very well be further stimulated by the potential success of the Voice Web, but they will be worked out within the larger context of the entire World Wide Web: HTML, WML, VoiceXML, etc.

[201] See http://www.w3.org/PICS/.
[202] See http://www.w3.org/P3P/.

Chapter 14. Structural Analysis – Speech Gateway Technologies

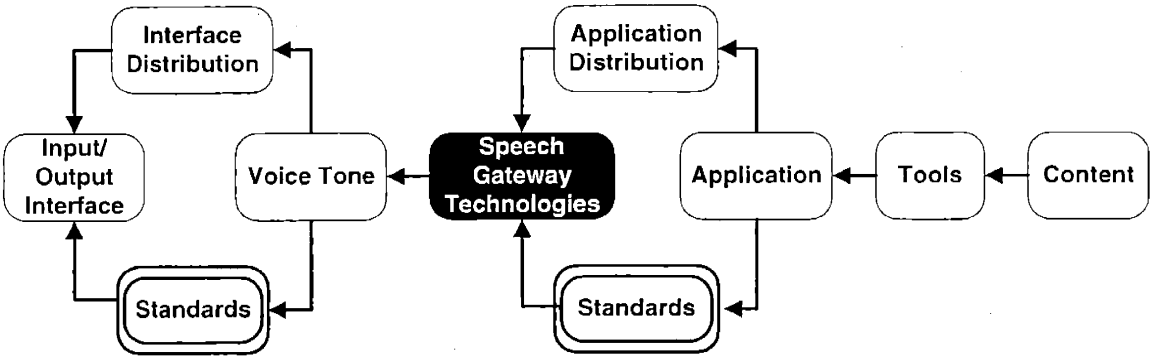

Figure 57. Market Structure (Currently on Speech Gateway Technologies)

14.1. Density

Once the novelty of speech-interaction has worn thin for the prospective Voice Tone Provider, their attention will turn to the evident disparity in density between traditional HTML and VoiceXML architectures. While the most dense speech gateway systems are designed to accommodate almost 800 ports (i.e. simultaneous callers) on a single telco rack, problems with scalability in the VoiceXML interpreter have kept this number out of reach in practice, and this density is an order of magnitude less than an HTML server using similar hardware. For example, one speech gateway vendor's specifications indicate that a single high-end server can accommodate 48 ports with a full complement of ASR, VoiceXML interpreter, telephony/data interfaces, and static audio resources; whereas that same server can easily accommodate 200 or more web clients actively surfing through content: a ratio of at least 4-to-1.[203]

While comparing circuit-switched vs. packet-switched architecture capacities is the textbook example of apples versus oranges, this issue of density does impact the bottom line of the service provider, particularly since the back-end infrastructure is so similar. VoiceXML's economy is based in part on its ability to reuse existing data serving architectures for the HTML Web. Suppose that a single XML application server generates simple HTML and VoiceXML documents at the same rate.

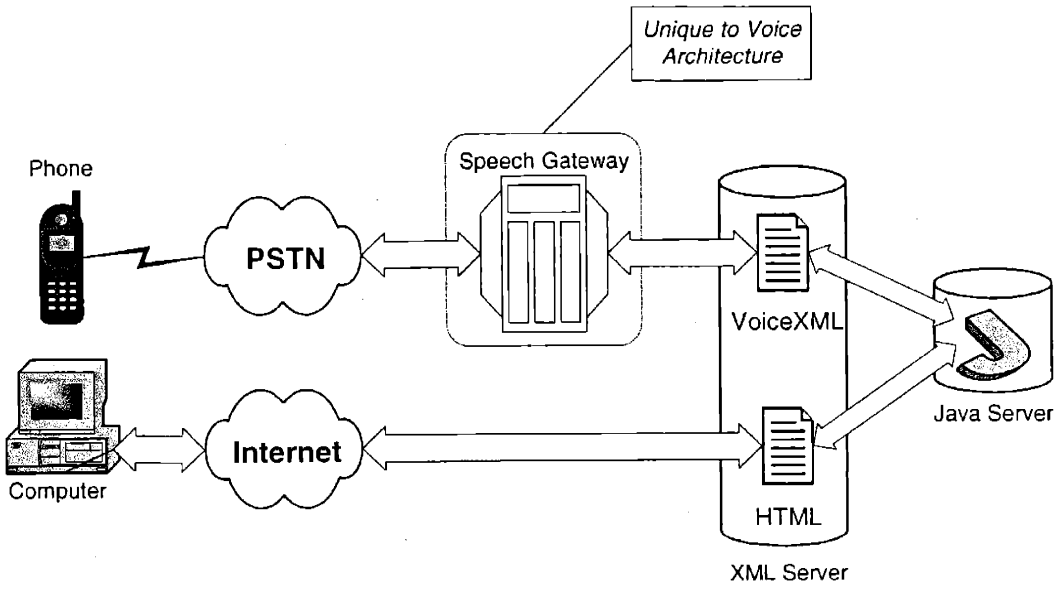

Figure 58. Voice vs. HTML Web Architecture

The HTML documents can be served directly to the users' browser, whereas the VoiceXML documents must first be interpreted and processed locally before leaving the system as audio. In other words, the difference between *consumer-side* and *service provider-side interpretation* is not simply a technology curiosity, since the additional burden of interpretation virtually guarantees that the voice architecture will always exhibit inferior cost density ($/port) and physical density (ports/server or ports/rack.) As the number of elements multiplies per live caller, so too does the complexity of operations and management.

It should be stressed that *scalability* is not equivalent to *density*: the former implies the ability to chain self-similar elements together to form a larger functional whole, whereas the latter connotes a throughput metric of functional performance. Any system intended to support a mass media phonecasting application should certainly exhibit the basic virtue of scalability, but the more troublesome metric of physical density increases the complexity of operations management and impacts the bottom line.

Fortunately, even if there were no efficiency improvements in the core algorithms and logic in the coming years, density will be improved as a natural result of the gains in general purpose processors, hard drives,

[203] The challenge of sizing an HTML server is both an art and science, but the *capacity planning methodology* attempts to minimize the startup risk for the initial deployment. For an excellent overview of this approach, see: http://www.webtechniques.com/archives/1999/04/menasce/ and http://patrick.net/wpt/ch02.html.

etc.[204] Especially as the potential market for voice applications grows, ASR and TTS technologies continue to be the subject of vigorous research at academic and corporate facilities. Information theorists are forever finding new ways to compress audio with minimal loss in fidelity, and today's streaming technologies are limited more by the vagaries of the public Internet than any inherent restrictions. Even the port densities of circuit-oriented telephony card profit from basic advances in ASIC and DSP technologies, independent of unlikely advances in the established PSTN network.

Apart from the anticipated advances in the basic building blocks of network technologies, the speech gateway vendor should still exercise parsimony in its implementations and excise any superfluous functionality. For example, many speech gateway vendors support VoiceXML functionality as a "feature" of a much larger middleware system that has been built up over many years in the enterprise IVR space. Instead of creating a multi-threaded (i.e. composed of 'light-weight' processes) browser that supports fast context-switching for a large number of sessions, the entire enterprise environment (or some large subset thereof) is replicated for every port. Apart from carrying the burden of unutilized, non-VoiceXML blocks of code, the browser is removed from the hardware by endless levels of unnecessary abstraction.

Lest the newer generation of speech gateway vendors point a derisive finger at the enterprise vendors, it should be noted that they, too, are guilty of extending functionality beyond VoiceXML. For example, the creators of the Nuance *Voyager* product anticipated the need for a browser (and many other interface elements) on top of their core VoiceXML interpreter, so they added a proprietary User Interface module to support navigation, searching, profiling, bookmarks, login, help, etc.[205] In addition, Voyager extends support to its existing proprietary ASR modules called Speech Objects, carrying yet another layer of complexity along for the ride. Despite the benefit of a clean slate, the resulting product is similarly bloated. It should be noted, however, that in response to its customers' demands for a streamlined interpreter, Nuance responded by producing a VoiceXML-only version of the interpreter called *FireWalker*.

In addition to technology advances and intelligent implementation of individual components, the speech gateway's internal architecture should also be configured for efficiency. As has already been suggested,

---

[204] For example, *Moore's Law* (transistor density doubles every 18-24 months) was first observed by Gordon Moore in 1965, and has been a remarkably accurate prognosticator of the exponential gains in processor density and many other fields.

[205] The intent of the Voyager example is not to malign Nuance's inclusion of the additional user interface functionality; to the contrary, they correctly anticipated many of the Human Factors issues that would prevent the creation of a true Voice Web.

the speech gateway need not be implemented as a single box but as the functional sum of individual resource pools. Just as non-trivial websites span multiple boxes (e.g. HTML server, Java server, database), a speech gateway can garner multiplexing gains by pulling together load-balanced collections of ASR, TTS, Audio, and VoiceXML resources. In a sense, each caller is assigned an instance of a virtual gateway, elements of which are fixed for the duration of the session (e.g. port) and others that are allocated on demand (e.g. ASR.) This schema avoids the inevitable underutilization of resources on any one box, as well as the dreaded upper limit of resources inherent in 1-box designs. These multiplexing gains increase in direct proportion to the heterogeneity of the application set: the VTP's worst nightmare is an overloaded box surrounded by underutilized boxes. Another advantage of the pooling approach is its innate flexibility and fault-tolerance: e.g. taking elements online (or offline) in a "hot" system, wholesale ejection of a vendor's technology in favor of another's, etc. It should be noted that these resource pools could be implemented in either special-purpose hardware or software running on regular servers, but the latter option affords the aforementioned Internet-wide gains in server performance and the possibility of repurposing the servers. Even if the hardware-only solutions proffer superior density factors, density should not be maximized to the detriment of other considerations.

Before closing this discussion of density, the author would like to erase an increasingly popular misconception that the telephony interface alone is responsible for the cost-performance differential between the VoiceXML and HTML/WML architectures. There is no argument that the PSTN interface would not profit greatly from the abandonment of TDM-switching, but it overlooks the other resource-intensive elements of the speech gateway: ASR, TTS, VoiceXML interpretation, Audio playback. It should be remembered that since the telephony interface is implemented entirely in hardware (ostensibly to avoid stealing CPU cycles from the host), the speech gateway's **server** requirements are largely unaffected by interfacial improvements. In short, so long as interpretation remains within the service provider's walls, the phonecasting architecture will always exhibit inferior density characteristics.

14.2. Audio Playback

While a speech gateway vendor cannot reasonably conclude that streaming audio is unimportant for phonecasting clients, it can certainly make it a lower priority or choose instead to focus on the audio-subdued enterprise hosting market. Even though the decision to support streaming is made simpler by the availability of development kits from the vendors directly, the integration task is still considerable. For example, some speech gateway platforms assume that all audio files are actually short prompts, so that the

file can be transferred in its entirety from local disk storage to the audio card (typically integrated onto the telephony card) before the start of playback; however, this schema does not yield acceptable performance for files of long (let alone unbounded) duration. Streaming assumes that the platform already supports the notion of a virtual file, parts of which can be separated and in transit at any given moment, e.g. the RealAudio Player assumes that the hard drive can stream the decoded data directly to the SoundBlaster's buffer across the computer bus. For example, Tellme's speech gateway architecture supports audio streaming of static audio files from across its internal network but does not currently support any streaming audio formats such as ASF or RealAudio; however, there is apparently no fatal flaw built into the system that would prevent the commercial audio systems from being integrated into this more basic network streaming service. Speech gateways that fail to implement real-time buffering internally for static files will be hard-pressed to overlay streaming technologies. Another potential streaming hazard is the failure to support barge-in during audio playback: apart from not having to compel the user to listen to anything she doesn't want to, imagine her frustration at being unable to exit from a live, continuous feed of a radio station. Clearly, building support for streaming audio is not a simple matter of downloading the SDK, but involves an evaluation of all the system resources from a **functional** viewpoint and making the necessary changes.

As for the proxy audio servers that are required to improve performance on both pre-recorded and live content, they are not technically part of the speech gateway (though an integral part of the VTP network) and were discussed at length in the *Application Distribution* Section.

14.3. Carrier-Class Speech Gateway

In the absence of a formal definition, the term *carrier-class* has become synonymous with "We make good stuff, so please buy some." The issue is whether the speech gateway (and by extension, its constituent components) is capable of sustaining the functional and operational reliability of a traditional telephony service. The term does not suggest a bias against non-carrier organizations, but does suggest some of aspects of a system required for large-scale phonecasting deployments.

- **Continuous Availability**. (It doesn't matter to me whether you decide to start these sections with or without capital letters, however, whichever you choose – they should be consistent.)Very low planned and unplanned system downtime, combined with effective overload control and graceful shedding of call-processing capacity in the event of component failures; hot-upgrade of system functionality and components on a live system

- **Robustness**. Redundant hardware, processor nodes, and communications interfaces in an always-on, all-active configuration; power sources with backups
- **Scalability (and Density.)** Ability to deploy systems and applications with low initial capacity, functionality and entry-level costs, and ramp capacity and functionality over time to high levels by adding hardware and software components in low-cost increments, while maintaining system availability
- **Manageability**. Provisioning of simple, efficient and effective mechanisms that enable the carrier's technical and human infrastructure to operate, administer and maintain the system's hardware and applications at low cost, extensive support for remote management of unstaffed facilities; audit automation; etc.

In a carrier facility, the platform would need to pass rigorous NEBS standards that check for physical and electromagnetic parameters, such as earthquake resilience and thermal robustness-NEBS compliance is the *de facto* standard for equipment operating within carrier networks.[206]

Needless to say, slapping a VoiceXML interpreter onto a computer and calling it "carrier-class" is a far cry from the intent of the term. Given the very high quality of telephone service in the United States, it would be difficult to imagine that consumers would not expect a similar quality of experience from phonecasting. Mere functionality is challenging enough for a speech gateway, let alone service levels approaching traditional telephony, but this is exactly what is required for large-scale deployments. Whether they consider themselves carriers or not, VTPs recognize that first impressions matter, so they will be reticent to incorporate platforms suitable only for enterprise deployments, and will likely forgo large-scale deployments until such systems appear. The technology reality is that VoiceXML-compliance alone is a challenge for speech gateway vendors, so it may take some time before they evolve into carrier-class systems; and while many of the gateway components predate VoiceXML, they have been infrequently deployed to support call volumes in the hundreds of thousands.

14.4. Component Economics

Apart from the engineering challenge of creating a scalable, dense, carrier-class, audio-streaming capable system, speech gateway vendors must contend with the more mundane but equally important issue of

[206] For more information about the Network Equipment Building Standard created by Bellcore (now Telcordia), see http://www.arcelect.com/NEBS.htm.

economics. Specifically, given the wide range of technologies that comprise a typical speech gateway, the vendor's integration role involves examining the cost structure for constituent technologies. Though it will be the Voice Tone Provider that will ultimately bear the cost of the system, the vendor must be ever cognizant of the price that the market is willing to bear for a speech gateway system, and strategically trim features to meet that clearing point.

ASR and TTS. Part of the challenge is to get these component vendors to recognize that a Voice Web market model would fundamentally shift the scale of the market, and the licensing models should shift appropriately. For example, most ASR licenses are structured as one-time per-port[207] costs, e.g. charges can range from $1000 (under 100 ports total) down to $500 (over 10K ports total) per port. Note that this type of license divorces ASR pricing from any sort of utilization metric, biasing the system against certain types of applications. Suppose two phonecasters each have 100-port capacity speech gateway systems: one provides a choose-and-listen audiobook service and the other supplies a highly-interactive role-playing game. The former will require considerably less ASR horsepower than the latter, but both service providers pay the same for ASR capacity. It certainly isn't the case that each ASR engine is uniquely assigned to each port-nearly all commercial systems assign ASR horsepower from a pool of resources to each utterance on a real-time, load-balanced basis. The per-port licensing is a legacy of the enterprise era, which exhibits high application homogeneity both within and across enterprises. The current licensing model for ASR and TTS technologies may not be sustainable in a world where application heterogeneity will become the norm and not the exception.

Audio. If the intended set of applications has no requirement for external audio or live streams-a reasonable assumption for most enterprise applications-then the speech gateway vendor's typically limited support for audio formats is not an issue. Even the increasingly popular MP3 format for music content is reasonably achievable with existing IVR audio technologies. However, a phonecasting service that employs branded content will typically need a playback engine that can cache and decode the stream in real-time, so the speech gateway provider must consider approaching one of the two major streaming audio vendors: Microsoft (Advanced Streaming Format) and RealNetworks (RealAudio format.)

The matter of licensing the relevant SDKs from these two vendors may prove challenging. The key is to understand the basic business model that these two companies adhere to for their streaming technologies: popularize the standard by giving away free basic Players to end consumers, and then charge content

[207] A *port* is logically equivalent to a single voice channel on the telephony interface, e.g. a T1 provides 24 *ports* of capacity.

providers for the Producer and Server components (RealNetworks also has a strong hosting presence called the Real Broadcast Network), and the end consumer can also purchase a premium version of the Player. Since each streaming technology provider relies on the fact that it is the sole provider of serving/production technology for its proprietary format and transfer protocol, it will avail itself of every opportunity to extend its share of this essentially duopolistic market-the last thing that RealNetworks and Microsoft want to become is commodity audio technology providers.

Since users ultimately drive the adoption of serving/production technologies, the Player (the piece of the system that the consumer direct engages) presents a natural opportunity for extending consumer awareness of the audio brand. For example, when a RealAudio file is encountered for the first time on the HTML Web, the user is directed to the RealNetworks site itself to download the player, which displays revenue-generating ads from a variety of external sites and tenders an offer to purchase the premium RealPlayer. Once the Player is downloaded, it will take the liberty of becoming the default audio/video player for many other popular formats unless specifically instructed by the consumer to the contrary. Every time the Player is run, the technology provider's logo appears, and a list of other broadcast content available on the Internet is prominently displayed at all times. Much more than a piece of engineering, the Player is a large part of the marketing arm of the audio streaming vendor.

All of these traditional revenue and marketing avenues depend upon the consumer's participation vis-à-vis *consumer-side interpretation* (i.e. the browser with which the Player interfaces is on the user's computer), an assumption that is invalidated by the *service provider-side interpretation* configuration of the Voice Tone architecture. From the caller's perspective, the interaction with the phonecasting system results in audio, the format and protocol of which should be completely transparent. Short of requiring an "audio vendor identification tone" each time a stream is engaged, the streaming technology provider must look to the licensing structure to recover the lost revenue and branding opportunities. Since most audio hosting/streaming contracts are tied to actual usage, the structure of the license would likely include an element of audio traffic capacity. The speech gateway integrator may elect to recover the licensing costs through a combination of the platform price and passing the audio costs directly to the VTP. In short, a speech gateway vendor has to ensure that there is sufficient demand among its target clients for external audio support to justify undertaking this licensing exercise.

The lengths to which the major streaming companies will go to protect their hold over their technology domains cannot be underestimated. They vigorously surround their protocols and formats with intellectual

property protection and pursue infringing firms, however small. For example, Streambox's *Ripper* product transcoded RealMedia streams into other formats. In December 1999, RealNetworks brought federal suit against Streambox under the Digital Millennium Copyright Act, the case being settled out of court in RealNetwork's favor on 8 September 2000.[208] The author of VirtualDub, an open source (freeware) video capture program that allowed users to convert from ASF into a number of other formats, was pressured into dropping this feature in the face of legal pressure from Microsoft.[209] This is not a matter of protecting each company's conversion tool sales since neither Microsoft nor RealNetworks has a tool that converts from its own format to another-why would they jeopardize their dominance in the audio industry? Once content is converted into streaming format, it is intentionally impossible to recover the static file. The relevance to the speech gateway integrator is clear: any attempt to develop a gateway that relies on conversion from ASF/RealAudio to some other traditional, internal format will be met with strong resistance and possibly legal action.

Telephony Interface. The telephony interface suffers from a similar pricing problem: The telephony cards are sold in fixed-port configurations (e.g. T1, Dual-T1, Quad-T1), and each port is uniquely assigned to serving a single call (an artifact of the PSTN's circuit-switched connection model.) Since resources are assigned irrespective of actual telephony load (i.e. speech event frequency and duration), application designs that discourage frequent interaction do not yield cost savings for the service provider. So while the ASR/TTS engines suffer from a *contractual* inability to multiplex, the telephone interface cannot multiplex at a *technology* level. By contrast, the data interface is a true packet-switched resource, so statistical multiplexing gains can be appreciated.

Apart from the particulars of the license structure or inability to multiplex, the sheer scale of these per-port costs can be quite daunting, especially in relation to the cost per customer for a typical HTML hosting facility. Consider the following costs[210]:

ASR = $500/port

TTS (high-quality) = $200/port

Telephony Interface = $150/port

Total = $850/port

[208] For more information: http://www.realnetworks.com/company/pressroom/pr/2000/streambox.html.
[209] For more information: http://advogato.org/article/101.html.

[210] These costs are actually aggressive given the author's pricing data for large installations.

Remember that this $850 doesn't include audio playback, the VoiceXML interpreter, or any of the many other hardware and software elements in the Voice Tone architecture. "Apples and oranges" disclaimers aside: consider that a standard $10,000 RAID server can service approximately 200 HTML users with acceptable responsiveness, resulting in a $50/concurrent-client ("port") cost that significantly undercuts the fixed-allocation voice elements alone. While there will doubtless be those that question the validity of these numbers and protest the comparison to an HTML "cost per port" metric, it should be obvious to the reader that there is a significant economic drag on the VTP's revenue model (the gateway vendor will simply pass on the costs) in supporting this burden of VoiceXML interpretation.

14.5. Elimination of the Circuit-Switched Telephony Interface

The gateway vendor can negotiate more creatively with the ASR, TTS, and audio software vendors; and it can pursue the most economic path to creating the software-based execution environment (e.g. VoiceXML interpreter, middleware, etc.), but the most obvious element that gateway vendors and VTPs alike would like to excise is the circuit-switched interface to the PSTN. In contrast to the cost-effective, efficient, commodity data interface on the opposite end of the gateway, the telephony interface card leaves much to be desired both in terms of one-time costs and the recurring costs of telephony connectivity.

Not all Packets are IP. The PSTN, not unlike the Internet, is a logical network comprised of many technologies. Most residential and enterprise users encounter telephony as POTS and T-carrier links, respectively, but there a variety of network technologies that the carriers employ within their own networks and for larger customers. For example, Asynchronous Transfer Mode (ATM) is a cell-based[211] technology that carriers routinely deploy throughout their core networks to carry both data and voice traffic without the inefficiency of T-carrier time-division multiplexing (TDM.) The common misconception that packet networks are unable to support telephony-grade voice overlooks the fact that the core of every major carrier network is already packet-switched: only the traditional "last-mile" elements rely on circuit-switching. However, whereas ATM was designed specifically to support cell-switched multimedia services through QoS provisioning, IP was not. So the current inability of IP-based networks to support QoS both within and across organizational boundaries should not lead the reader to conclude that packet-switched networks cannot support telephony. To the contrary, in addition to its role

[211] A *cell* is simply a packet of fixed-size.

within the carrier network, ATM or SONET/SDH is routinely employed at inter-carrier traffic exchanges, and large enterprises already have the option of ATM adapters to support traditional call center functions.

The fact that all speech-aware telephony cards are currently designed to support only POTS or T1 service levels is a vestige of its enterprise origins: since a company would rarely need to support thousands of simultaneous callers, multiple T1 interfaces would be sufficient for the expected load. However, a mass market product like phonecasting may require tens or hundreds of thousands of ports, which would be difficult to manage if the incoming link always had to broken down into a torrent of DS-1 (24-port) increments. The current state of the art for PSTN card, 4 T1s per card (i.e. quad-span T1), certainly lowers the cost per port but does nothing to eliminate the inflexibility of the TDM-based PSTN interface.

The obvious suggestion would be to marshal the existing ATM products targeted at the call center services and repurpose them for speech-interactive applications. While this would certainly enforce packet-switching from carrier to VTP, the reality is that speech-interactive cards require functionality beyond mere telephony. For example, speech processing requires echo cancellation, voice energy detection, prompt termination, speech event signaling, audio signal buffering-all of which would be expensive to do in software, especially in real-time.[212] The current crop of ATM (or similar telephony switching technology) adapter products would have to be endowed with these features before integration with the speech gateway platform. Since the ubiquity of phonecasting services is mere speculation and most speech applications require fewer than 96 ports, the development of speech-appropriate ATM cards will be driven only by the realized growth of the large-scale voice industry. The point is simply this: packet-switched telephony is already available to large organizations, and the evolution of the telephony interface to attach directly to these networks (and not to their inefficient appendages) will allows vendors to avail themselves of statistical multiplexing gains.

Voice over IP Gateways. The theoretical appeal of extending the carrier packet-telephony technologies to the customer (i.e. VSP) premises must be tempered with reality: there are currently no high-capacity cards tuned for speech-interactivity, and the ATM-to-the-enterprise movement seems to have stalled before it even started.[213] Meanwhile, the growth of IP-based networks to accommodate the growth of the

[212] IBM's DirectTalk VoiceXML platform uses regular telephony cards and software to enable these speech-specific features. The viability of this architecture for a large-scale VoiceXML platform remains to be seen. From all accounts, specialized telephony boards are the *status quo* of Voice Tone networks.

[213] Parallel to the demand for large-scale VSP services, there is an emerging multi-service trend that uses ATM to dynamically assign the bandwidth of a single DSL pipe to voice, data, video, etc. These ATM-to-the-premises projects were big news in the past few years, but they have amounted to little more than media hype. Examples

Internet has spurred academic and corporate research to discover means by which real-time services (notably voice and video) can be conveyed over the non-QoS public Internet.[214] For both technical and economic reasons (e.g. how to price service levels across network boundaries), there is no generally available QoS technology deployed and employed throughout the public Internet; however, this does not preclude the use of IP-voice technologies within networks, especially enterprise and service provider intranets.

As the name would suggest, a Voice over IP (VoIP) gateway mates the PSTN (typically T1 links) to the intranet (typically Ethernet) and enforces switching policies that ensure QoS sufficient for carrying real-time voice, usually over dedicated network segments unburdened by regular data traffic. In effect, a virtual telephony card interface is distributed across the VoIP and speech gateways.

include: Sprint ION (http://ion.sprint.com), WorldCom On Net (http://www.worldcom.com/for_your_business/on_net/), AT&T INC (http://www.ipservices.att.com/products/).

[214] Refer to the Internet Engineering Task Force's Transport Working Group to sample the various and sundry suggestions for creating a QoS-aware Internet: http://www.ietf.org/html.charters/wg-dir.html#Transport_Area.

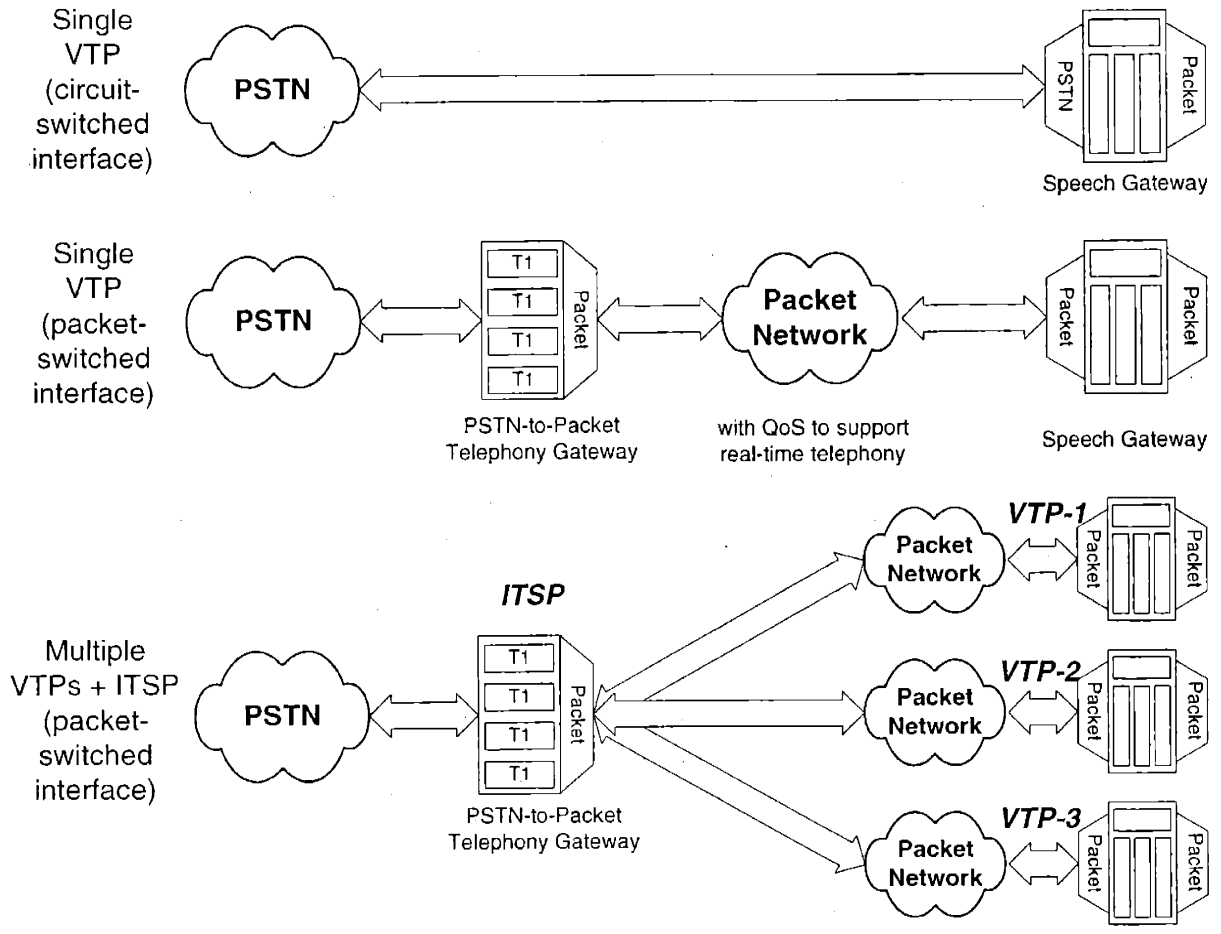

Figure 59. Evolution of Packet-Switched Telephony Architecture

The advantage to be gained by moving to a packet-based telephony interface is not immediately clear from the standpoint of a single Voice Tone Provider. The speech gateway itself may no longer rely on inefficient circuit-switched technology directly, but the Voice Tone network is still saddled with it: the cards are simply shifted within its own network from the speech gateway to the VoIP gateway, which must still be populated with expensive, TDM-based telephony cards. The VTP must also continue to endure the high recurring rates of traditional telephony since the same external carrier links are used.

The technological and economic gains can be appreciated, however, if the operational responsibility for the VoIP gateway is moved outside the VTP's organization; specifically, if an external Internet Telephony Service Provider (ITSP) were to provide each VTP a dedicated data connection from its central packet telephony network. Since the ITSP market services multiple VTPs and many other kinds of telephony business, it experiences economies of scale in switching and PSTN interfacing on behalf of all

its clients, which it uses in turn to undercut traditional telephony rates. This configuration exactly mirrors that of the traditional relationship between Internet Service Providers (ISPs) and Web Hosting firms: the former is concerned about providing the highest reliability in sustained connectivity, and the latter is concerned with supporting the functionality that its application clients demand.

Whether or not a viable ITSP market exists as well as the degree to which cost efficiencies can be realized will be the subject of the Interface Distribution section. However, there may still be strategic value in anticipating this network evolution even if the costs of both the VoIP and speech gateways must be borne by the VTP directly. In the absence of an IP-based telephony supplier, the VTP will be gaining invaluable knowledge about how to negotiate the challenges of ASR over packet-based networks. Intranet network segments, even if they are dedicated to telephony traffic, do not behave as deterministic channels as traditional TDM links do, so there will necessarily be a learning process of maintaining speech gateway accuracy and performance over an on-deterministic network. Once that process has been completed to the satisfaction of the VTP, it can begin to incrementally outsource its internal VoIP operations to external ITSP agencies without incurring undue service risk. The VTP that attempts a wholesale changeover of both its telephony services supplier and its basic telephony network model may be unpleasantly surprised by the complexity of the transition.

If and when the ITSP market materializes, the cost structure of the other speech gateway component technologies must be reexamined. The telephony interface is the last component of the speech gateway to retain any concept of a "port," so ASR and TTS vendors will have to evolve their pricing policies accordingly, e.g. how can ASR licenses be granted on a port basis if there are no ports or anything else that retains a one-to-one relationship with the callers. Perhaps they will resort to per-CPU pricing, licensing on the number of unique customer records in the back-end database, total ASR "throughput" per month, etc. The effect on audio licensing is unclear given the lack of precedent for incorporating audio streaming technology.

Chapter 15. Structural Analysis – Voice Tone

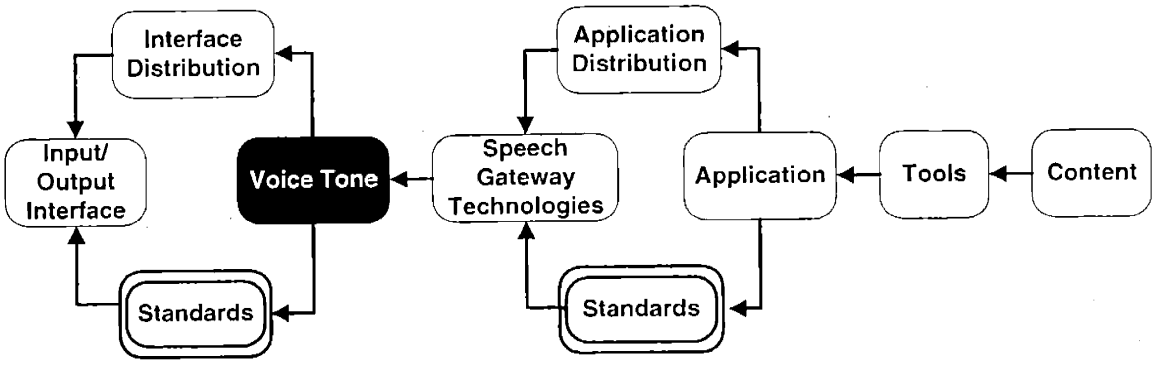

Figure 60. Market Structure (Currently on Voice Tone)

VoiceXML was designed specifically for a client-server architecture, so many of architectural components (and even staffing needs) are the same as those employed in a traditional HTML hosting facility: servers, mounting racks, emergency power backup, on-site operations staff, element management systems, security, cabling, routers, databases, remote backup and monitoring, etc. However, the responsibility of operating the speech gateway and the incoming lines requires *call center* skills as well: redundant trunking, SS7 and ISDN signaling, switch management, call detail records and settlement invoicing, etc. Given that personnel with both telco and data office skills are difficult to find along with the general risk of equipment startup costs, many phonecasters and even carriers will outsource the voice tone network altogether, patterned on the now-common practice of outsourced web hosting.[215]

There are already a handful of companies that provide hosted VoiceXML services as a standard product. Some have a long history in traditional IVR and call center management, while more recent entrants have showcase consumer portals through which they hope to drive market share for hosting services. Unlike a solitary phonecaster that would be burdened with an underutilized architecture during off-peak periods, the VTP can reap economies of scope by multiplexing its clients' traffic across all its ports. Of course, there is a price-premium for shifting the underutilization risk into the hands of the VTP, but that margin will fall as the voice application market grows.[216]

---

[215] The creators of VoiceXML anticipated the existence of this hosting market: "For example, one option would be to contract with a service provider until the voice application had proven its worth, and later purchase a VoiceXML platform to own and operate." VoiceXML Technical Background, http://www.voicexml.org/tech_bkgrnd.html.

[216] As of the writing of this paper, a monthly per port price under $200 (not including incoming minutes) would be uncommon, even for high volumes.

The status of the Voice Tone Provider is a precarious one within this evolving market, since it is the most likely target of vertical integration. Most firms in this market combine this element with other market segments, rendering the term *Voice Service Provider* almost useless. The content delivery network operators (e.g. Akamai) may be increasingly successful in recruiting away the hosting side of the business to which IVR hosting firms have grown so accustomed, so the VSP is left only with speech gateway side of the business. But then the gateway side of the business may be co-opted by the carriers (e.g. Sprint) themselves in their effort to capture cellular market share. It may be the case that the voice tone element is vertically integrated into the consumer portal industry just because all VoiceXML content flows through these AOL-like networks (e.g. Tellme.) The gateway vendors themselves may offer subsidized VTP services to drive the adoption of their equipment (e.g. Voice Genie.) The voice tone element may even evolve into just another piece of software on the caller's cellular phone given sufficient time for Moore's Law to work its magic. The result is that it is difficult to talk about the economic factors that drive the VTP market segment in isolation from the other segments, other than the trivial observation that it would profit greatly from any cost reductions in the speech gateway equipment itself.

So as not to steal its thunder, the consideration of the VTP segment as the wild card in a dynamic voice market will be reserved for the final chapter.

Chapter 16. Structural Analysis – Interface Distribution, Input/Output Interface, & Standards

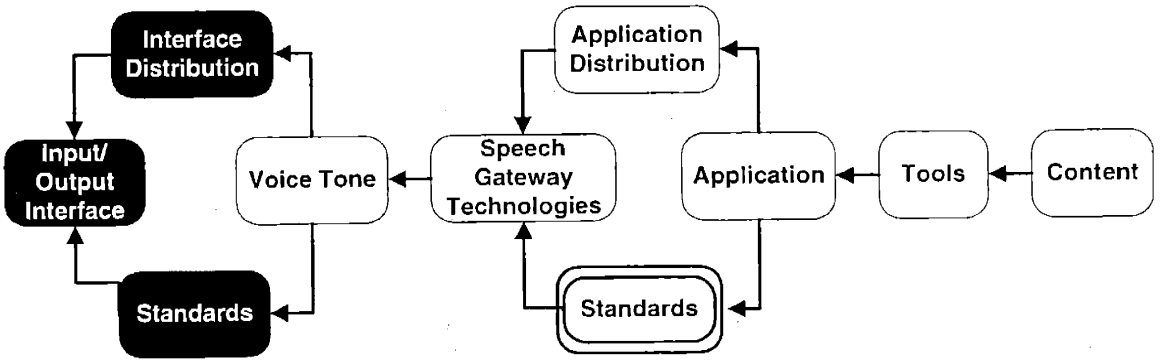

Figure 61. Market Structure (Currently on Interface, Distribution & Standards)

Just as it would be difficult to talk about a keyboard or mouse without also discussing the wires and protocols that interface them to the computer, the end user device is integrally tied to the network from whence the content comes. It may seem odd to devote serious attention to the VoiceXML network equivalent of the mouse and mouse cord, but this is in fact the essence of the distribution. The telephone is a dumb audio send/receive device that is directly manipulating the VoiceXML browser, albeit at much larger distances than a keyboard would be from its host computer.

16.1. Justifying the Telephone Assumption

Before moving forward with the discussion of the network itself, the presumption that the regular wireline or wireless telephone is the best choice for delivering phonecasting services should be laid bare. A review of the elements of the speech gateway and the larger Voice Tone network reveals few elements that are dependent upon or limited to using the regular telephone. With the obvious exception of the telephony interface card in the speech gateway, the rest of the voice tone network is fully capable of delivering a generalized experience over any audio network. For example, the I/O interface may take the form of a microphone and speaker attached to the gateway directly for the purpose of speaker-dependent dictation on a personal computer. Next generation CDMA networks hold the promise of higher bandwidth and media players integrated into the cellular phones themselves, opening the possibility for a phonecasting

experience with full stereo-quality output.[217] Users may even opt for mobile audiovisual devices like the 3Com Palm and WindowsCE-compatible devices that already have the technology to connect to wireless networks, albeit at speeds slower than real-time stereo streaming allows. Given the natural fit between commuting and non-visual interactive services, there are variety of vertical players focused on addressing that captive market, including the next generation of the OnStar in-dash system[218] and the CommandAudio on-demand interactive system that utilizes unused radio spectrum to deliver high-quality audio to a custom consumer device.[219]

The purpose of enumerating these competing and often superior interface technologies is to illustrate not only that there is indeed a migration path beyond telephony for "phonecasting" services, but also the barriers created by these innovate interfaces. Note that these interface upgrades suggest upgrades to the entire cellular network, the ubiquitous deployment of QoS services throughout the Internet, and even the creation of entirely new distribution networks distinct from the telephone network. The cost of these end user devices themselves is a barrier in and of itself, particularly if the interface has no other purpose than engaging the Voice Web.At the same time, no one has ventured to estimate what the connection costs or pricing schemes will be for the next generation cellular network.

While many of these technologies will indeed factor into the future development of the Voice Web and the voice industry in general, there has been little doubt that the telephone, whether wireless or wireline, is the appropriate I/O interface for the foreseeable future. Despite the limitations (e.g. 8-kHz monophonic audio) of the telephone network in the face of the voice tone network's capacity to deliver rich media services, the telephone has been blessed as "canonical" device for voice services from the very beginning. The precursors to the speech gateway industry (e.g. IVR) were forged in the delivery of interactive call center services, so all the elements of the speech gateway are geared to this end: the ASR engine is specially tuned to filter out noise on cellular networks, the TTS and Audio engines are optimized for 8-kHz output, etc. The VoiceXML and DialogML specifications extend support for a variety of telephone-specific features, e.g. call transfer, hang-up, etc.

---

[217] The Samsung Uproar is a regular dual band-dual mode cellular telephone that also has a built-in MP3 player. Unlike the next generation devices envisioned by the 3G movement, this phone does not play MP3 streams real-time from the network: sounds are downloaded at regular WAP speeds for future playback. See http://www.samsungusa.com/.

[218] See http://www.onstar.com/.

[219] See http://www.comamndaudio.com/.

While it appears that the telephone's blessing as the target I/O interface was a forgone conclusion, there are indeed factors at work that suggest that the telephone is indeed the most appropriate end user device, regardless of the limitations that it imposes on the consumer experience. Perhaps the most compelling argument is that the telephone network is the largest (in terms of end user devices and traffic) electronic network both in the United States and especially abroad. However compelling other interface technologies may be, they would be hard-pressed to offset the market penetration and installed user base advantages of the simple telephone. Consumers are already familiar and comfortable with the telephone, and most importantly, they do not need to spend a large sum just to join the phonecasting network. Even if a cellular phone purchase is being considered, phonecasting is but one of many services that the user can access through this single device, the cost of which is subsidized by the wireless carriers to reduce the barrier to entry.[220]

Apart from the consumer drivers above, the other market players are solidly behind the telephony interface. Carriers forever in search of augmenting their commodity wares with value-added services have a vested interest in perpetuating the cellular interface, particularly after subsidizing the cost of the phone for each new subscriber. The telecommunications equipment firms certainly don't object to the projected growth in the speech gateway sales and the attendant increase in general infrastructure revenues-though the handset folks may prefer technologies like WAP that require users to constantly upgrade their phones. Assuming that the technology barriers to the Voice Web are suitably addressed, then the potential consumer market (i.e. anyone with a phone) provides strong justification for undertaking VoiceXML development. Given the already impressive growth in the cellular handsets over the past several years, the voice industry would indeed be remiss if it were to forgo the positive network externality and scale economies of the existing telephone network.

So for the purposes of this paper, the interface distribution discussion will be limited to traditional telephony networks and devices, with the understanding that much of the analysis also applies to the alternatives listed above. Unless the voice tone provider intends on building its own telephony network from scratch, it relies on the services of other wireline and wireless carriers-notable exception: the VTP is an established carrier. If the application provider makes the determination that the recurring cost of the call is to be borne by the caller (no toll-free access number), then the major concern is the sizing of the

[220] Side note: In some ways, the cell phone is a captive device of the carrier because each is specifically programmed to work on a particular carrier's network. For example, even though the Motorola CDMA phones made for Verizon Wireless and SprintPCS employ the same technology and work on the same bands, particular factory settings ensure incompatibility. If a user wants to switch providers, they have to buy a new phone-the hidden cost of subsidization.

incoming trunk to manage call blocking expectations. However, if the provider elects to stimulate consumer demand by offering subsidized access in part (flat monthly subscription for unlimited use) or in whole (e.g. toll-free number, carrier-specific 'star'- or 'pound'-key sequence on wireless phones), then the VTP is intimately concerned about reducing recurring telephony costs, especially if the application requires *bridging* (i.e. connect and hold) transfers and conferencing services.

16.2. Whither Packet Telephony?

In addition to the cost efficiency of a packet-switched telephony interface, the VTP also strives to reduce the recurring usage costs on each call.[221] The typical suggestion is to enlist the services of an *Internet Telephony Service Provider* (ITSP) that can deliver the call in IP-format (not simply one that employs Internet telephony within its own network.)[222] ITSPs typically employ a network of private IP links to eliminate the degradations in service commonly experienced with telephony over the Internet at large. Given the much-publicized cost-efficiency of packet voice over that of circuit-switched telephony, one might expect that ITSPs would offer a significant discount to the prevailing long-distance rates; however, just like any player in a commodity market, the pricing strategy is to undercut the prevailing price, not to work up from the base cost. Whatever the supposed savings in transport, PSTN arbitrage is the pricing model of choice for ITSPs. Consider the following samples of residential long-distance pricing:[223]

- Sprint Sense AnyTime (PSTN): 10¢/minute, $4.95/month (subscription fee)
- Sprint Nickel AnyTime (PSTN): 5¢/minute, $8.95/month
- Qwest (ITSP): 5¢/minute, $4.95/month (online invoice) or $7.95/month (direct bill)
- Global Crossing Exact Rate (ITSP): 7¢/minute, $1.99/month (online) or $3.99/month (direct)
- GTC Telecom (ITSP): 5¢/minute, $0/month (credit card) or $1.95/month (personal check)

While the degree to which the ITSP companies have propagated IP telephony within their networks cannot be gauged exactly (e.g. they may rely extensively on reselling wholesale minutes from larger carriers), the anticipated sizeable cost reduction is certainly not evidenced by the pricing samples above. This does not imply that the VTP will pay rates comparable to the residential rates listed above-traditional IVR hosting firms can routinely provide sub-4¢/minute pricing because of their high volumes-but it does

[221] Barring an exceptionally close relationship (e.g. co-investment) with a telephony carrier, in-bound toll-free pricing under $0.03/minute reflects aggressive discounts for high volume commitments.
[222] While the term *ITSP* biases the discussion to IP-telephony, the arguments apply equally well to any other packet telephony technologies, e.g. Frame Relay, ATM, etc.
[223] Pricing information was collected directly from the websites of the companies listed on 15 January 2001.

suggest that the distribution of the phonecasting service over a packet-based telephony network may not yield significant cost savings, at least in the short-term. In fact, more ITSP firms are interested on arbitraging the historically high settlement rates established by the International Telecommunications Union[224] in other countries, rather than competing for the low margin telephony business in the United States. Perhaps the most colorful summary of the ITSP market's failure to materialize is suggested by the title of former FCC Chairman William E. Kennard's speech at a recent conference: *Internet Telephony – America is Waiting.*[225]

The reader should bear in mind that this criticism of the ITSP industry does not imply that IP telephony cannot be utilized within the VTP's own network, but it does suggest that such a move would be more strategic than economical in the short term. [226] The importance of reducing the recurring telephony costs cannot be overstated, since all marginal revenue must be tempered by this marginal drag. Even if **all** one-time infrastructure costs could be reduced to zero, there could **still** be a 3¢ or 4¢ dead weight attached to every minute the caller is engaging the system. The per-minute cost is multiplied for phonecasters that offer multi-link services like conferencing. If phonecasting is intended to be a free consumer service, then the service provider needs to come up with very aggressive revenue sources, virtually eliminating any chance of a sustainable advertiser-supported business model.

16.3. Forward-Looking vs. Sunk Costs.

The *carrier* is unique in that it is the only voice tone provider that can leverage its own telephony resources. When a carrier builds a **wireline** telephony/data network, the usual process involves a high one-time cost for trenching and laying conduit, so the carrier will put as much fiber or copper capacity in that link as possible: the cost of populating the pipe with "dark" strands is but a fraction of the "digging" costs. Particularly for long-haul (i.e. inter-LATA) networks, the issue of capacity is not a matter of laying more fiber but of turning up more switches. Whether the carrier employs SONET, ATM, or even IP technologies to carry telephony traffic through its internal network, the per-minute cost to itself is certainly lower than the lowest nationally advertised price. The market price for telephony reflects

---

[224] See http://www.itu.int/.
[225] Voice Over Net Conference on 12 September 2000. See http://www.fcc.gov/Speeches/Kennard/2000/spwek019.html.
[226] In December 2000, David Reich of IBM illustrates a VTP architecture that employs a telephony/VoIP gateway, but there is no mention of the organizations that can provide this service outside the intranet. See http://www-106.ibm.com/developerworks/library/ibm-voicex.

markups for advertising and branding, various and sundry fees imposed by the FCC (e.g. SLC, PICC, USF)[227], access charges assessed by the originating and terminating LECs[228], customer service and billing, customer churn, etc. While there is no documented source that reveals the actual basic trunking and switching costs for telephony, telecommunications experts and telecom models[229] suggest that the cost is close to 1¢/minute, even with traditional PSTN technologies.

If this assumption is correct, then given equivalent service models, a national wireline carrier has a potentially insurmountable cost advantage over a standalone phonecaster. This advantage is not limited to only the network, but also to all those previously-discussed call center elements required to support a voice tone network (e.g. SS7 signaling, CDRs, redundancy), all of which the carrier has already propagated throughout its network. In some cases, the carrier already has a business unit dedicated to traditional IVR hosting (e.g. WorldCom), so the deployment cost of VoiceXML over a preexisting network may be a fraction of that for the build-out of a greenfield voice tone network. In other words, the deployment costs for interface distribution (and the voice tone network itself, though less frequently) are not seen as *forward-looking*, but as *sunk*-an investment already made for another service (the investment may also already be recovered, especially for older telephony networks.)

Residential long-distance is an example of a service that avails itself of sunk-cost network elements. Circuit-switched telephony networks are built to meet peak traffic demand, which normally occurs during the business day. Given *business* voice traffic is more profitable on a per-minute basis than *residential* traffic (e.g. higher per-minute rates, lower customer churn, fewer calls placed to customer service, fewer "deadbeats"), carriers will design their wireline networks to meet the demand of business traffic. Once the workday is over, however, the carrier can either consider the underutilized network as a cost of doing business, or as a revenue opportunity by offering residential long-distance. Despite the apparent "inferiority" of residential traffic, the cost of serving them is simply the marginal cost of customer service and billing since the network and switching equipment are already deployed. In other words, residential traffic is not a new business that requires an entirely new build-out, but leverages an existing investment for another product.

---

[227] See http://www.fcc.gov/Bureaus/Common_Carrier/Factsheets/telephone_bills_facts.html.
[228] See http://www.fcc.gov/Bureaus/Common_Carrier/Factsheets/ispfact.html.
[229] Examples include: HAI Model (nee Hatfield Model, http://www.hainc.com/model), Hybrid Cost Proxy Model (see http://www.fcc.gov/ccb/apd/hcpm/), Telecom Economic Cost Model (see http://www.microeconomics.com/model/costmodel.htm)

Given that many phonecasting applications are targeted to the mobile consumer, it is important to identify similar sunk-cost elements that can be marshaled from a **wireless** carrier's existing infrastructure to facilitate distribution. The wireless access network between the consumers' handsets and the cell towers is a clearly distinguishing feature of a cellular network, but the network that actually ties these cells together closely resembles a typical wireline network. Given that almost every major cellular provider in the United States is owned by a large wireline carrier, it is safe to assume that these wireless carriers enjoy fixed-line network economies between cell sites.[230]

While there is little concern for bandwidth limitation between cells, the tower and spectrum capacities may present an issue for certain types of voice applications. Once a carrier has secured through public auction a spectrum license from the US government, it deploys cell towers in strategic locations around the service area to meet peak demand in high-traffic areas (e.g. a major intersection that gets gridlocked with traffic every work day.) Depending on the cellular technology adopted by the carrier, the number of concurrent calls that a single cell can handle can range from only 56 (AMPS) to around 600 (CDMA.) If a single tower's load is causing an unacceptable number of blocked calls, the carrier has the option of creating adjacent cells or replacing the cellular technology itself: depending on the logistics, the first option may not be available and the second potentially requires an upgrade of the entire network.[231] Since each cell install may cost upwards of $1 million and encounter recurring costs for leasing space, maintenance, electricity, security, and traffic backhaul; the mobile carrier would like to keep marginal infrastructure expansions to a minimum.[232] Unlike a wireline network that can absorb increases in traffic at low marginal cost, the wireless access portion of the mobile network exhibits high marginal costs of expansion.

Given this cost factor, certain types of phonecasting applications may actually **negatively** impact the carrier's revenue flow. Perhaps the worst-case scenario for a mobile carrier is an application that supports long-play content and reaches peak usage during commute intervals, e.g. listening to live World Cup action from Europe during the long drive home. As the adoption of this proposed service would grow, so too would the call hold times increase, requiring the carrier to increase cell capacity to simply maintain

[230] Particularly for remotely situated cell towers, the carrier may opt for a microwave link back to the Mobile Telephony Switching Office (MTSO) instead of the more common T-carrier.
[231] Theoretically, the carrier also has the option of leasing additional spectrum from the government, but the intermittent auction process in the US removes this option from consideration as a short-term solution.
[232] For a general introduction to the facets of building a cellular network, see http://www.geckobeach.com/cellular/articles/wireless.htm

the minimum acceptable blocking rates. This application presents a situation diametrically opposite from the residential long-distance example earlier:

Application	Residential Long Distance	Long-play audio
Sunk Cost Element	Long-haul network for daytime business traffic	Cellular network with traffic peaks during morning and afternoon commutes
Temporal Overlap	Very little overlap, since residential traffic occurs during the evening	Almost 100% overlap
Distribution & Peak	Demand generally spread out through entire evening, peak load smaller than peak business load	Severe peaks during commute hours
Target audience	Residential consumers who are distinct from business customers	Existing cellular customers
Marginal expansion costs	Limited to customer service and billing	Cost of new cell site and related recurring costs
New Equipment	None, since residential and business telephony share the same switches and links	Requires voice tone network, which likely doesn't exist in the carrier network *a priori*
Opportunity Costs	The network is underutilized at night, so existing business not threatened	Application benefit must offset the annoyance of existing or even **new** customers who encounter busy signals more frequently

For these reasons, cellular carriers may forgo a sustained deployment until the capacity constraint issue is addressed.[233] Of course, the application under consideration may exhibit an entirely different usage characteristic, e.g. short-play interactive audio with uniform load distribution throughout the day. It may also be the case that the absolute magnitude of the phonecasting-related traffic load is only a small fraction of the total traffic, but that fact alone may suggest that there is insufficient demand to warrant investment in the voice space. The only way to confidently predict the traffic pattern is to commission trials in limited markets, but early anecdotal accounts and the author's personal experience indicate that even short-play or interactive content traffic coalesces around the commute hours. The point is not to suggest the appropriate application for a wireless carrier, but to further motivate a comprehensive *economic analysis*, i.e. contrary to the emphatic assertions of corporate press releases and the doting media, it's not just a matter of whether the application is "nifty."

[233] The role of next-generation wireless technologies like 3G will be discussed in the Technology Factors section.

16.4. Bucket Pricing Model.

A brief survey of the pricing models for cellular service (and increasingly, residential long-distance) illustrates a strong predilection for *bucket pricing*: a monthly fee for any number of airtime minutes up to an upper bound. The genius of this model is that consumers *feel* as though they are receiving a low per-minute rate, e.g. a $30 plan that includes 300 airtime minutes is only 10¢/minute... **if** you use exactly 300 minutes. If a consumer fails to utilize all the minutes in her plan she is actually paying a higher-rate per minute; and usage over the contract limit incurs sizeable per-minute charges. Either way, the carrier invariably "wins," especially since the bucket plans are specifically designed to encourage *under-consumption* in the long-term.[234]

Unlike a per-minute model in which a carrier makes money only when a call is made, a bucket model implies that the carrier would actually prefer that the subscribers never actually use their phones. Since their monthly revenue stream is relatively fixed for existing customers, every minute of use is taking money out of the carrier's wallet.[235] This does not imply that the carrier doesn't have to offer a high quality of service-to the contrary, it appreciates the intense competition for consumers-but it only needs to provide stellar service when the subscriber is actually using the phone.

Against this backdrop of bucket pricing, one might question whether a carrier would actually appreciate the potential influx of minutes due to phonecasting applications. If a subscriber maintains her existing calling plan, but then uses additional minutes for phonecasting without exceeding her monthly limit, then the carrier is clearly the loser: remember that each minute on the system cuts into profits. This is similar to the situation caused by the introduction of WAP service: how should the service be priced within the context of an airtime bucket model? Some carriers charge a monthly fee to allow the user to pull WAP minutes from the total bucket of minutes, while others assess a monthly fee for a separate bucket of minutes. Regardless of the particular model, carriers are very protective of their minutes in a bucket framework, so the value of **any** service that actually **encourages** phone usage is carefully weighed against the "opportunity revenue" of bucket underutilization.

---

[234] And potential *over-consumption* in the short-term as the subscriber attempts to find his cellular "sweet spot." Side note: The author's monthly usage is consistently around 300 minutes, but his cellular provider's "closest" plans are buckets of 180 and 400 minutes. Coincidence, or something more?

[235] Just think of cellular carriers like health clubs: the model is to get you to sign a contract and then hope you don't show up too often.

Chapter 17. Structural Analysis – Marketing

Though absent in the stylized version of the Market Reference Model, the Marketing and End User segments were mentioned at the beginning of this chapter. This section explores the role of Marketing the phonecasting product to the consumer, either directly or through intermediaries.

During the past several years of Internet IPO mania, there had been a definite shift from business-to-consumer (B2C) to business-to-business (B2B) products and revenue models, motivated by the recognition that the road to win consumer loyalty is long and expensive. At the same time, there are a select few that have sustained their B2C orientation and have actually thrived even in the midst of the recent economic downturn. There are obviously many deterministic and random variables that affect any firm's success, but at the risk of making sweeping generalizations, those that have fared best exhibit at least one of the following: consumer products that create *network externalities*, and *adjunct* products that create value for an existing product. The relevance to marketing and branding costs is clear: the former (network externality) suggests the possibility of a vertical integration strategy, whereas the latter (adjunct) advocates a co-partnership strategy with existing consumer firms.

Consumer Products with Network Externalities. A product for which the utility of the good increases with the number agents consuming the good is said to exhibit positive network externalities.[236] A classic example is the personal decision to buy a Windows or Macintosh computer: even though the Mac may be more appealing to the user, the idea of being limited in terms of programs and interaction with fellow computer users causes many to err on the side of popularity. The Internet protocol and public IP network are another example: as more sites are put on the web, more people will want to surf, which creates a larger audience, causing new firms to create a website, etc. Network externalities can indeed be created by the creation of an open standard or framework, but the Windows example above indicates that this need not be the case. Even AOL's hegemony of the non-WWW access market demonstrates network externalities: given the existence of a ubiquitous and open Internet, the market is willing to tolerate at most 1 firm that requires special document formatting conventions.

So the relevant issue is whether the phonecasting application under consideration creates compelling network externalities for consumers, which is obviously a larger strategic issue than simply the manner of

[236] For an excellent overview of the various types and degrees of network externalities, see http://www.utdallas.edu/~liebowit/jep.html.

marketing and branding. Perhaps the product is designed similarly to AOL in that it promotes proprietary keywords that will be prominently displayed alongside the HTML Web URL and the AOL keyword. Another cue from AOL is the creation of live chat rooms (essentially topic-oriented party lines) or a dating service. Whatever the particular externality, it creates an opportunity to appeal directly to the consumer to build momentum in the adoption stream. Clearly, a commodity weather/news/stocks/etc. audio portal does **not** exhibit a network externality, since a particular caller's utility is uncorrelated to the number of other subscribers. In the absence of any meaningful network externalities, perhaps the only factor that would warrant a direct-to-consumer marketing model is a compelling *first-mover advantage*.[237]

Adjunct Products. For the commodity service above, perhaps the more appropriate marketing decision is to disintegrate that role entirely. In particular, a phonecasting firm with a commodity product may decide to offer its services as a adjunct to other consumer firms, the most likely candidates being wireless carriers and dominant network portals (e.g. Yahoo!, AOL.) Both of these types of firms already possess broad product lines that are distributed through well-established marketing and branding channels, and they view the addition of voice functionality as a natural part of the competitive process for their relevant markets.

- **Wireless**. Regardless of the utility of WAP services, for example, few leading cellular providers would risk being the only firm without such capability. Given that the acquisition and loyalty costs for each subscriber range from $200 to $400[238], the carrier is willing to expand its service offerings to gain an advantage in recruitment and retention: a marginal phonecasting cost of $5/user/month may not seem like so much money in light of potentially losing a customer after a short time. Given the sizeable cost of cellular network build out, carriers already spend marketing dollars to populate the system with as many bucket-plan subscribers as possible, and keep them on the network as long as possible (even if that means spending a few extra dollars each month.) They even use airtime minutes as the "currency" for customer loyalty through programs that actually encourage usage, such as free evening or weekend minutes, calling circles, referral incentives, etc. Airtime minutes may cut into the bottom line for bucket pricing, but the loss of a customer cuts deeper.

---

[237] For example, Yahoo captured the lion's share of the WWW portal market through first-mover advantage and incremental service upgrades. However, a firm that fails to back up a first mover advantage with a network externality can often be easily displaced, e.g. Microsoft's Windows dominates the desktop computer GUI market pioneered by Apple's Macintosh.

[238] See http://www.sponsorship.com/forum/wireless.html.

Phonecasting may be perceived a natural adjunct to existing cellular services: voice mail, text messaging, WAP. Whether it is provided as part of the standard bucket-plan or an additional plan, cellular providers may view voice applications as the latest move in the strategic war for consumer awareness.[239] For example, SprintPCS has already launched a rudimentary speech-driven application called Voice Command, which affords a dial-by-name service for an additional monthly fee of $10. In conjunction with the advantage of lower *distribution* costs discussed in the previous section, this seems like a natural model for undifferentiated phonecasting services, one that extends the cellular industry's long tradition of outsourcing or contracting for voicemail and other adjunct services. The traditional wireline carrier market will also participate in this market by default through its almost complete ownership of the cellular industry, but it may also be active in its own right, provided that a phonecasting application that appeals to wireline user can be conceived.

- **Dominant Portals**. Given that most phonecasting applications are targeted to mobile consumers, becoming a phonecasting adjunct to a cellular firm seems like a natural pairing; however, portals like AOL (proprietary) and Yahoo (WWW) also bring considerable market power to the table. Given that portals offer essentially all of the same services (e.g. email, chat, shopping, etc.), the role of advertising and branding in securing market share cannot be overstated. Unlike the carrier arrangement, portals view phonecasting as an additional access point to existing content and applications. For example, AOL Time Warner commands an impressive array of consumer products and services, so the telephone is a natural extension of the distribution network, e.g. CNN starts out as a traditional cable broadcast, then gets ported to the HTML site, then the WML site, etc. These dominant portals also enjoy "self-branding" economies, e.g. Home Box Office can advertise an upcoming event on the WB Television Network (both of which are AOL Time Warner properties.) While the major portals lack the cellular distribution networks of the carriers, their close proximity to content (and their ability to restrict licensing to competitors) provides a similar cost advantage. The portals' existing marketing and advertising machinery can be extended to support a phonecasting product at very **low** marginal cost.

In short, portals have a clear content advantage and carriers have a distinct distribution advantage-both enjoy marketing economies of scope and scale that permit these organizations to promote a new product at very low cost. A phonecasting firm that decides to go head-to-head with these large players in a

[239] The issue whether WAP services are valuable to consumers may be moot, in that the *perception* of functionality may be valuable from a strategic marketing perspective. This war of feature one-upmanship is typical of a market

marketing war for the consumer must possess unassailably unique and useful products that create compelling network externalities.[240]

that provides a highly commoditized consumer service (wireless telephony).

[240] On 21 June 1999, startup Talk2.com took out the following full page national ad in the Wall Street Journal: "We're simply serving fair warning that, by September, we will begin beta testing a viral-marketing Internet business model from which we expect an initial 10,000-user base to grow itself to over half-a-million users in just one year. We expect to be profitable in the first two quarters of operation, and anticipate blasting Yahoo!, Lycos and Infoseek completely off the planet by 2001. Have a nice day." Several observations: 1. They don't have any user base because they evolved into a technology firm (**adjunct** carrier product) 2. They are not profitable. 3.Yahoo! is still the #1 HTML Web portal, Lycos merged with Terra to form a multinational media firm, and Infoseek is the search technology behind all of Disney's destination sites (e.g. ESPN.com, ABC.com, etc.).

Chapter 18. Structural Analysis – End User

The often-ignored yet all-powerful arbiter of the phonecasting market's success is the consumer, yet most callers have no strong preconceptions of what a Voice Web is or the types of applications that can be reachable through it. As with the HTML Web, it is ultimately the applications (i.e. sites) that will hopefully satisfy the needs unknown of the consumer. While there certainly isn't a cookbook recipe for creating successful Voice Web applications, the following factors should be thoughtfully considered before embarking on application design:

Beyond Y.A.W.N.S.S. The HTML Web's success relies in large part on the creativity of the destination sites themselves. It would be difficult to imagine the commercialization of the HTML Web based on a select few commodity services or products, yet this is the danger if the voice industry fails to eliminate the barriers to a Voice Web of peer destination sites. A cursory review of the existing consumer voice portals reveals an alarming homogenization of the application space, prompting the author to coin the acronym *YAWNSS*: Yet Another Weather, News, Sports, Stock service. (You have already defined YAWNSS and Wapathy in the footnote on page 106 – do you need to give the definition again?) The voice industry must already contend with the WAPathy (WAP apathy) engendered by early adopters of WML phones, so application providers must rise to the occasion to shed the dross of this perception.

Primary vs. Secondary Content Source. While most users are casually interested in new information services, few are actively looking for more services to subscribe to. For example, a person may be sincerely interested in a new magazine, but feels overwhelmed by the knowledge that she doesn't even have the time to read the ones she already subscribes to/gets (?). The HTML Web is a sufficiently large space such that users are besieged with multimedia information at work and at home, so an audio-only service in these contexts would compare quite unfavorably. However, there are moments during which people are visually engaged in a primary experience and would like to engage a secondary source, e.g. listening to talk radio during a long commute, playing music on a walkman while going for a run, etc. If the user requires access to interactive content or information during these moments of primary engagement, she cannot employ a service that requires a shift in her primary focus. In this regard, phonecasting's lack of a visual interface is its strongest asset in that it will be positioned as an ubiquitous secondary information source. This does not suggest that a phonecasting application cannot be a primary service, but that the general tenor of consumer voice services eschews the hyper-competitive primary space in favor of the untapped, mobile secondary market.

No Such Thing as Cellular Toll-Free. Since minutes are metered on the basis of airtime usage, toll-free numbers are counted against users' monthly totals on the cellular network. Even a very highly motivated phonecasting subscriber would be reluctant to burn through his monthly allotment of minutes in a matter of hours. For example, this caller may be genuinely interested in live sports feeds from soccer leagues in Europe, but not for 10¢/minute when the same feed can be accessed freely as part of the flat rate ISP package he already subscribes to-he can contain his enthusiasm until he gets to a computer. The essential problem is that while the wireline network can support general toll-free calls, most phonecasting applications will be targeted to the mobile consumer for whom the toll-free call is but a fond memory. Despite the potential attraction of on-demand, long-play content, the current bucket pricing model for minutes biases the user against it, so much so that it ultimately represses demand and thereby undermines the development of the phonecasting sector in general and the Voice Web in particular.

It should be noted that the cellular carrier has the option of offering free (i.e. not deducted against the airtime balance) minutes for any service, either to promote its own services (e.g. free calls to customer service and account info) or through custom partnerships with other firms (e.g. listeners of WSJZ 96.9 FM Talk Radio in Boston can dial #969 on any Verizon phone for a free call to the radio station.) Provided that the business model could sustain toll-free access for its subscribers, an individual or portal phonecaster could theoretically strike deals with all the major carriers for special 'pound' or 'star' key sequences. However, most carriers would co-opt this service model for themselves if they felt this was a viable business model.

The Voice Web is Different. VoiceXML's pedigree would initially suggest that creating a voice application is a natural extension of the normal XML design process; however, the voice programmer soon appreciates that the process is driven more by unique Human Factors considerations of the user than the familiarity of HTML coding. For example, dealing with the real-time input of data within a client-server architecture is already a sizeable shift from the norm. The design team must also invest heavily in processes that are typically absent (though thoroughly applicable) in most HTML implementations: multiple rounds of consumer testing, context-sensitive and escalating levels of 'help' facilities, graceful recovery from input error, etc. While the advent of VoiceXML will certainly lower the barrier to publishing content on the Voice Web, an appreciation for the time-honored principles and proponents of user interface design is key to developing a successful phonecasting product.

Part of the business design process is the humility to recognize when there is simply not a good fit between the target application and the telephony interface. The Voice Web is not the "banana republic" of the HTML Web in which visual applications are cannibalized and made to fit the procrustean bed of the telephony; to the contrary, innovative application providers will view the mobile, audio-only interface as a distinct opportunity and not as a barrier.

This advice is no better expressed than by Dr. William Meisel:

> "The term *Voice Web* should not be understood as simply extending the Internet to conventional telephones—although it certainly does that. For example, viewing the impact of telephone speech recognition as merely making the Internet mobile—extending it to wireless telephones—misses the power it adds to any telephone. Telephone speech recognition provides a benefit whenever the telephone voice user interface is superior to Web browsers. Services that are easier to use over the telephone include getting quick information such as weather; buying something in response to a print or broadcast ad; or managing voice calls. The Web browser interface is a powerful paradigm, but when there are too many pages and complex navigation, it suffers from many of the deficiencies of touch-tone telephone applications.
>
> Telephone voice interactions are fundamentally different than visually-oriented Web browser interactions. Some things are done best by voice and others by a visual interface. The best businesses will understand these differences and take advantage of them, using synergy between the two when possible. *Voice Web* is a good term to summarize a major opportunity, but the analogy to the World Wide Web should not be used to mask the critical differences."[241]

[241] *The Voice Web and the Telephone VUI*, Editor's Notes (May 2000), William S. Meisel. See http://www.tmaa.com/voice_web.htm.

Chapter 19. SCENARIO ANALYSIS

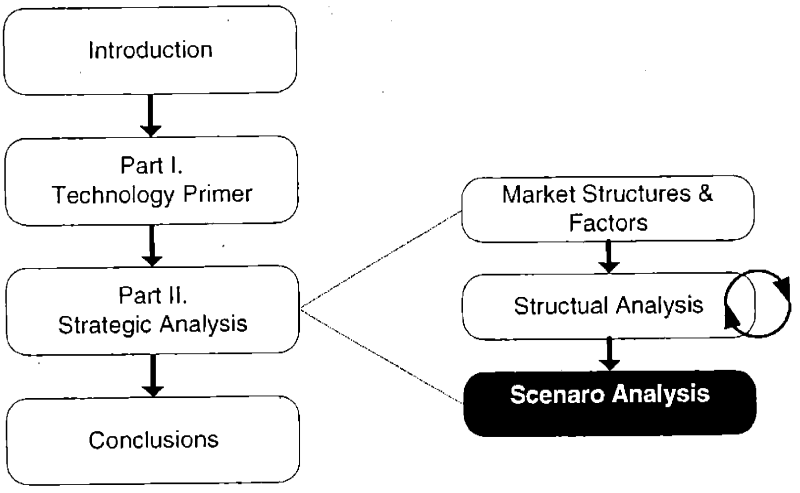

Figure 62. Overview of Paper (Currently on Scenario Analysis)

In the Technology Primer, the reader was introduced not only to the technology of the unfamiliar speech gateway, but also the characteristics of the elements that surround it, e.g. content delivery networks, VoIP, etc. In the previous chapters of Structural Analysis, this technological knowledge was marshaled alongside regulatory, economic, and human factors to enumerate their effects at each step in the value chain from application provider to end user. In this chapter, these factors are considered for their likely effect on the evolution of the voice market, always with an eye towards creating a true Voice Web.

Disclaimers. The author is fully aware that he is about to dabble in the dark art of prognostication, and fully admits that the outcome is not deterministically based on the factors outlined in the previous chapter. To the contrary, it is highly likely that many factors have been overly discounted or simply overlooked entirely. Contrary to academic assertion, players within any market will not necessarily act rationally to maximize their long-term welfare. X factors such as emotional connection to an outcome, single-minded focus on boosting quarterly financial reports, disdain for competitors, and mother-bear protection of dwindling revenue streams all contribute to the shaping of this voice market and are extremely difficult to account for in any analysis. The author could also be just plain wrong. In any event, what follows is a hopefully well-reasoned synthesis of the foregoing discussion and a starting place for discussion.

19.1. Fiction of the One-Model Market

For all the success of the HTML Web through all segments of publishing and technology sectors, it does not control 100% of the networked visual application market. AOL currently has over 27 million subscribers, 23 million more than it nearest competitor Earthlink. Despite all the technology and market arguments that could be made in the HTML Web's favor, AOL continues to enjoy great popularity with consumers who appreciate the seamless, intuitive interface and experience that it provides. The vertical content provider market has certainly evolved over the years, such as consolidation into a single player and opening up limited access to the HTML Web, but the walled garden approach is essentially unchanged. AOL also had the great foresight of building an enormous positive network externality through its chat rooms which continues to be a driving force for the service. Coupling these facts with its marketing reach through direct advertising and pre-install arrangements with computer manufacturers, an image of stability is warranted.

Similarly, despite the author's unabashed preference for the Voice Web architecture, the voice industry will most certainly have players that seek to replicate the vertical AOL model, including AOL itself through its AOL By Phone service. A single portal whose applications have been custom designed to work together seamlessly may have a distinct interface advantage over the peer-to-peer model of the Voice Web, particularly given the unfamiliarity with speech-driven applications in general. The cellular carriers may also take up the mantle of vertical integration given their direct relationship with the end consumers: why let callers leave the safety of the carrier's own voice portal, especially since they can exact rents for placements on the main menu and favorable keywords.

If the barriers for the Voice Web can be proactively eliminated, then the market will naturally follow the course to vertical disintegration, but not in every quarter. There will always be remnants of the market that will be able to sustain vertical integration because of first-mover advantage, marketing reach, network externalities, etc. However closed these initial consumers portals are, they serve an important role in raising the public consciousness about voice services. It also doesn't hurt that the public is cutting its teeth on consumer services that have been well-considered and set the standard for VoiceXML applications in the future. Vertical portals like Tellme and AOL By Phone are natural phenomena in the voice industry's evolution: their existence is not to be decried but to be anticipated so that avenues for more development are kept open. If the Voice Web fails to materialize, it will not be the fault of dominant vertical portals but rather the voice industry's for failing to build the foundation for the future peer network, e.g. Vocal DNS, grammar standardization, etc.

So in the Market Models that follow, the reader should keep in mind that these are not mutually exclusive views of the voice industry, nor are they intended to last indefinitely.

19.2. Short-Term Trends

19.2.1. Voice Tone Clearinghouse

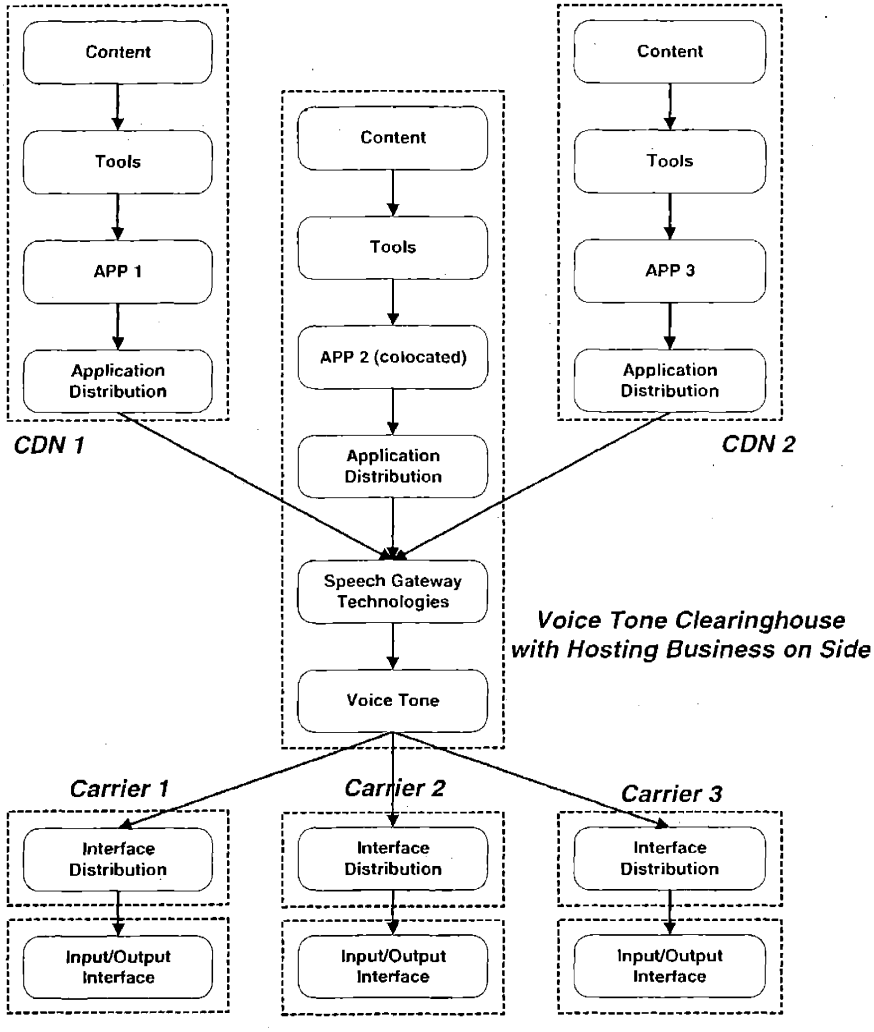

Figure 63. Voice Tone Clearinghouse

In the pre-HTML days of the Visual "Web," companies like AOL and CompuServe built extensive access networks so that their users could connect to the content servers. These POTS modem banks were wholly

owned and operated by a single content portal, and the only destination was the portal itself. As the HTML Web market began to develop and a more general need for modem access to ISPs emerged, AOL and its contemporaries divested their access networks entirely and received access services through the general market. This of course did not imply that AOL was interested in creating a more open content architecture, but that it wanted to focus its resources on building the AOL offerings and not the access networks that were rapidly becoming commodities. In the current market, there are access wholesalers (e.g. WorldCom) that do not market directly to the consumer but rather focus on providing access to the branded ISPs and vertical portals.

The role of the Voice Tone Provider may closely parallel that of the wholesale modem access market. *Voice Tone Clearinghouses* could compete for the wholesale business from a variety of sources, including carrier portals, branded consumer portals, and even individual Voice Web destinations. Like any access provider, the voice tone operator would have to guarantee a minimum level of uptime, reliability, redundancy, etc. Fixed access numbers would be assigned to each client, but all the incoming calls would be dynamically assigned to the pool of speech gateway resources. The content and applications (e.g. APP1 and APP3 in the picture) from media-centric sites would normally be hosted by and delivered to the Voice Tone Provider through QoS-ensured Content Delivery Networks such as Akamai and the Content Bridge Alliance. However, sites without a strong HTML media presence may opt for value-added hosting services that the VTP could provide (e.g. APP 2 in the diagram.)

This model very closely matches that of the traditional IVR hosting model, especially in those cases where the application is colocated on the VTP premises. Note that this model assumes strong standardization efforts have taken hold and that application portability is the norm and not the exception. Early VTPs may have sufficient risk-affinity to build various flavors of VoiceXML architectures to the detriment of economies of scale, but first-mover advantage plays an important role in the wholesale markets. This consumer model can be further augmented with revenues from traditional hosting services for enterprises in order to keep the lights on until the Voice Web materializes.

When it does materialize, the natural question is who will pay the VTP? In the HTML Web model, consumers pay a flat monthly rate for general Internet connectivity; by extension, would consumers pay a similar monthly rate for generalized access to the Voice Web? Unless their usage of the Voice Web approaches that of the HTML Web, most consumers would not be willing to pay very much per month, especially if several sites offer the services through a toll-free number. The voice tone clearinghouses may opt for a relationship whereby the destination site is assessed a per-minute fee for the total inbound

minutes, but smaller sites (e.g. personal or hobby sites) would not consider subsidizing access for casual surfers.

As the telephony and dialup networks continue to merge at the edge of the network, the VTP and the ISP may be increasingly difficult to distinguish, suggesting that generalized IP access for both the Voice and HTML Webs will become the standard. However, such a configuration is well into the future. In the short term, a combination of low monthly fee plus application subsidies may be the most clear route. Before the reader scoffs at the idea of actually paying for Voice Tone access, consider that consumers are generally willing to pay $10 per month to access the WML Web, SprintPCS's voice dialing service is also $10/month, and AOL will start charging $4.95 per month in April 2001 for subscriptions to AOL By Phone.[242] Given that access to Internet media has never been free, why should access to the Voice Web be an exception?

19.2.2. Voice Tone Network Integrated into Content Delivery Network

[242] See http://www.aol.com/anywhere/aolbyphone.adp.

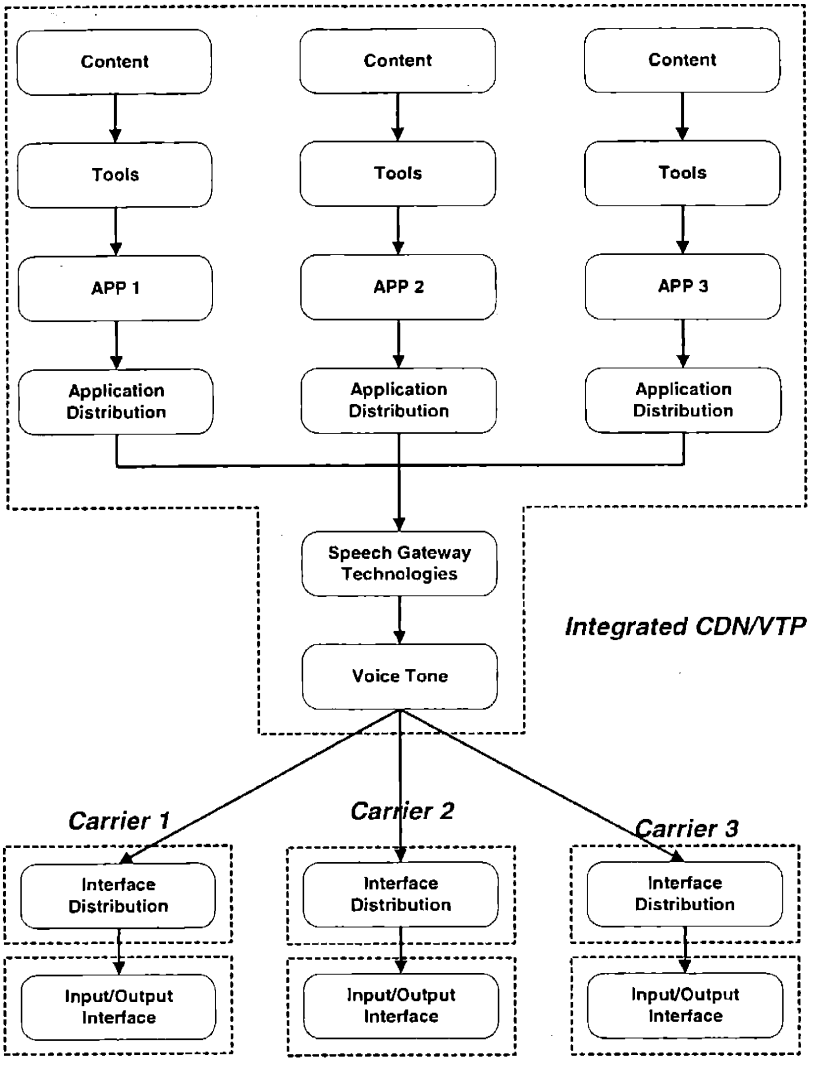

Figure 64. Integrated Voice Tone & Content Delivery Network

While traditional IVR firms and even Voice Tone clearinghouses will enjoy enterprise hosting revenues for quite some time, the shift to consumer media services, both on the HTML and Voice Webs, suggests each destination site will prefer to consolidate its media products into a central repository. For example, if CNN is interested in deploying its own VoiceXML and WML application that utilizes its existing audio and text content from its HTML site, it will prefer to centralize the content repository and generate the appropriate XML proximal to the data. In other words, it will definitely avoid situations where it must needlessly mirror content and manage multiple XML installations. The promise of XML is the ability to leverage back-end infrastructure, so why would CNN be eager to manage the complexity of another installation for the voice site? This trend is not specific to media-rich sites since regular HTML sites similarly enjoy the enhanced the performance and reliable hosting services of companies like Exodus and Digital Island.

Remembering that remote hosting of the voice application is possible only with the services of CDN operator, so they are in a unique position to move down the value chain. This would be a particularly attractive option for audio-centric voice applications, since a single contract with the CDN handles hosting, application distribution, and even speech gateway services. The CDN would consider this as yet another value-added service in its portfolio to eschew commoditization (e.g. caching, firewalls, monitoring, etc.) and would attract media clients away from other CDNs that do not offer the voice tone capability. The added media load on the CDN network would be marginal given that the same audio content is already being served over the network, so its internal caching and edge services would minimize the impact. The Voice Tone-capable CDN can also engage in wholesale VTP services for the more risk-averse CDN networks.

In order to mitigate the risk that Voice Web demand fails to materialize immediately, the CDN can buyout an existing IVR or VTP-centric firm and use the enterprise hosting revenues to carry the VTP network to the point of viability. This also has the marketing advantage of luring away the traditional HTML hosting business for firms with enterprise sites on the acquired hosting firm.

19.2.3. Voice Tone Network Integrated into Internet Telephony Service Provider

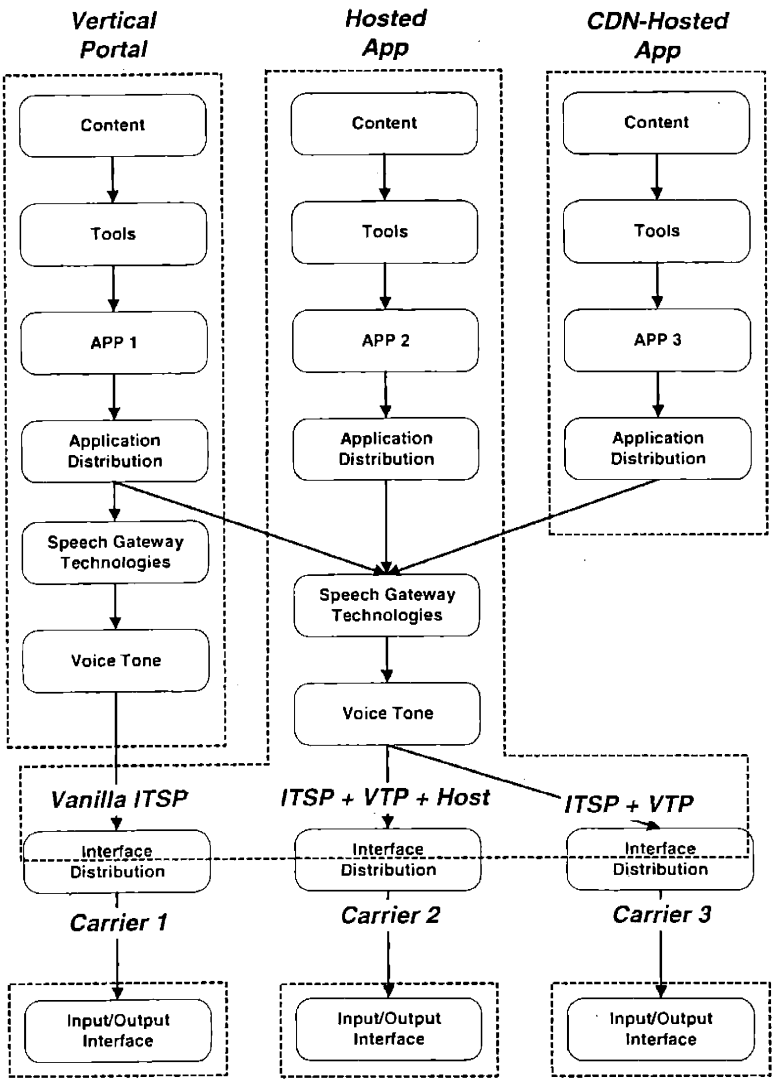

Figure 65. Integrated Voice Tone & Internet Telephony Network

Just as remote application hosting is possible only with Content Delivery Networks, significant reduction of both the one-time and recurring interface distribution costs is possible only with Internet Telephony Service Providers (or similar purveyor of packet telephony.) While there is little large-scale ITSP activity for US-based traffic, companies like IBasis have built successful business on arbitraging the international settlement rates: iBasis (which is the correct spelling: IBasis or iBasis?) terminates its private IP links on foreign soil and wholesales the telephone/data/fax service to local carriers and other resellers.

Vanilla ITSP. The most basic service that the ITSP can provide is simple IP telephony to VTP facilities, illustrated on the left-hand side of the diagram. The target customers may in turn wholesale the voice tone

service or the customer may be running a portal application of its own; in either case, the provider must have already upgraded its internal gateway interface to accept the incoming packet stream. The diagram above depicts the ITSP as integrating only part of the interface distribution network since it is unlikely that the entire call can be completed end-to-end over the private VoIP network: e.g. the "last mile" link may be a POTS line or a wireless POP. The recurring cost savings is a function of the ITSP's pricing strategy and its network footprint (i.e. the more traffic stays on network, the less it has to pay out to PSTN intermediaries.)

ITSP + VTP. This scenario is depicted on the right side of the diagram: the situation is similar to the previous, with the additional functionality of providing the voice tone service. This would be the case of a firm interested in deploying a voice application using audio content already hosted on a content delivery network; rather than incur the expense of running its own voice tone facilities, it outsources this functionality to the ITSP. In this case, the ITSP has a cost advantage over the "traditional" voice tone provider that is tethered to the PSTN, so it can potentially undercut the price of other VTP services.

ITSP + VTP + Hosting. This is the same service described above with the exception that the application is hosted directly on the ITSP's facilities. This offers most compelling value for applications that are hosted in the United States but have access numbers worldwide: the ITSP can trunk the call to the US at a fraction of the normal cost, and the call never leaves the ITSP network. It can also realize economies of scale by aggregating traffic from a variety of locations to a single voice tone facility as opposed to populating its entire network with voice tone gear.

While the advantage of the ITSP shines most clearly for traditional hosting services, it offers no real advantage for the Voice Web model. Firms that deploy media-rich VoiceXML applications will tend to already have relationships with CDNs that support their HTML applications, and they would rather connect to a voice tone network remotely than bother with creating a satellite facility. In anticipation of the possible evolution of the CDN industry to include voice tone functionality, the ITSP might preemptively strike by offering CDN services of its own, evolving into a multiservices IP carrier (as opposed to IP telephony only.) This effect could also be achieved through a strategic merger with an existing CDN operator.

In summary, the ITSP's strength lies in its reduction of voice trunking, but that advantage is offset by the more probable location of audio content off of the ITSP network. This does not imply that the enterprise hosting segment of the voice industry will not be lucrative-in fact the author would argue that an ITSP-

host has almost insuperable cost advantage over the traditional hosting firm-but its offers no real value to the Voice Web destination sites directly. It must either content itself with participating as a commodity IP carrier for the larger CDN market, or seek to enter that market itself through partnership or merger.

19.2.4. Vertical Portals (a.k.a. Walled Gardens)

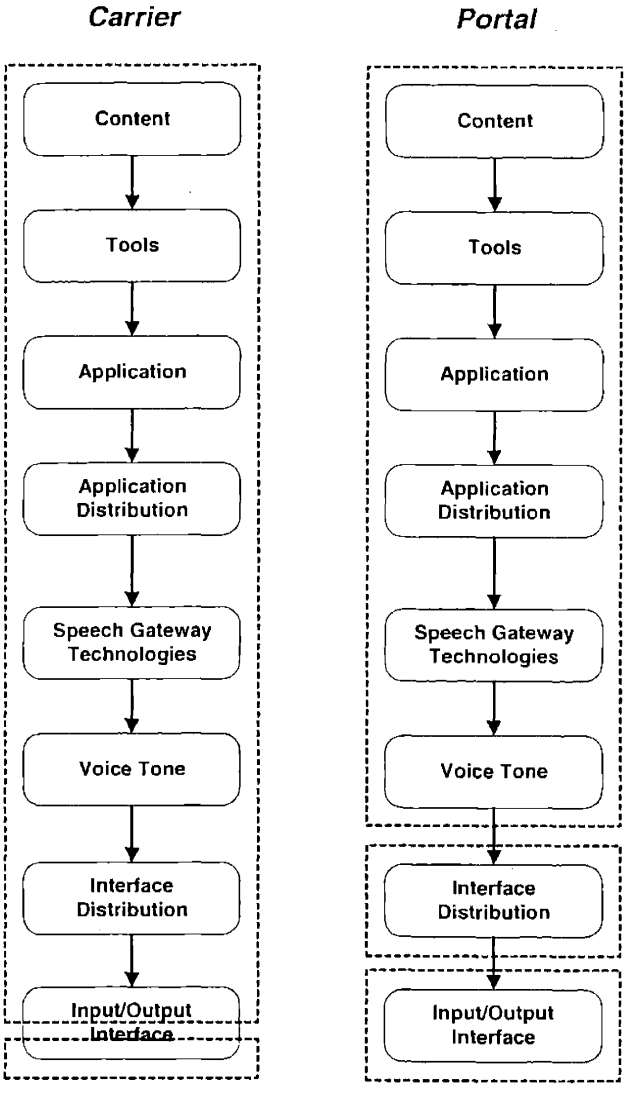

Figure 66. Vertical Portals

Carriers. The three previous models may actually be quite idealistic given the most common mode of access: cellular telephone. Unlike enterprise applications, the author assumes that consumer phonecasting

applications will be targeted primarily to mobile consumers-if so, then the wireless carriers are the most enviable position in the value chain.

Unless they are willing to buy a phone on another network, consumers who want WML browsing are captive customers of the carrier's WAP service; in contrast, the phonecasting consumer is free to call the number of any consumer portal or Voice Web entry point-the carrier's only role is to move the audio signals reliably. However, in the absence of a technological barrier, the cellular provider may erect economic and human factors barriers to ensure that users select the carrier's own vertical portal.

From a pricing perspective, the concept of a toll-free call on the cellular network is nonexistent (with few exceptions), so even if the Voice Web entry point or the alternative voice portal were willing to provide a toll-free number, it would provide no advantage to the mobile consumer. Unless consumers are so enamored with phonecasting that they abandon all notions of frugality, they will be reluctant to spend a large portion of their airtime minutes on getting weather/news/sports/stocks/etc. In contrast, the carrier's "house" portal can be a tool to sell another bucket of underutilized minutes, and it can set that bucket price flexibly to undercut the telephony airtime rate to increase customers loyalty and attract new customers. Note also that the I/O interface itself (i.e. the handset) is depicted as part of the carrier's network because of the high level of subsidization and the inability to use the phone on any other network. Perhaps most importantly, the carrier can extend its WAP practice of collecting high rents for inclusion and preferential placement on the portal's default menu.

Another economic advantage is the low cost of application and interface distribution. The former is near zero since the applications are colocated with the speech gateway (or through a dedicated trunking connection), and the latter is completely internalized within the carrier's own network. There is no need to rely on PSTN or even ITSPs to realize cost efficiencies.

From both a technological and human factors perspective, the carrier is apparently not obligated to even offer the possibility of accessing external VoiceXML sites. On the WAP browser, that feature is well obfuscated in the third page of the main menu and the user has to fumble through the DTMF keys to enter a long URI, but at least the possibility remains. The portal's applications, even from third parties, will most likely be colocated with the speech gateway on the carrier's network to ensure quality of service-in the absence of a CDN connection to the carrier, the latency between the remote site and the speech gateway will likely destroy any remaining desire to connect to external content. Finally, to be completely fair to the carriers, the Voice Web doesn't exist yet, so there is no content to access even if the incentives

could be aligned properly. The carriers, like the other vertical portals, will be the proving ground for phonecasting applications.

Independent Portal. Given that carrier portals exhibit striking efficiencies among many dimensions, one may legitimately question the viability of an external audio portal. First, the top spots on the carriers' portals may very well be these independent portals, e.g. AOL, Yahoo, and MSN Mobile occupy the 3 of the top four spots on the author's WAP browser. Second, whereas the carrier portals can uniquely leverage their existing network infrastructure, the independent portals can leverage their existing content networks.

For example, the newly merged AOL Time Warner commands an impressive portfolio of broadcast and interactive properties: MapQuest, MovieFone, CNN, the WB network, HBO, AOL Chat, etc. In order to consolidate its leadership position, AOL Time Warner may elect to reserve exclusive rights of telephone distribution for its own content. While there are often suitable competitors to each of AOL's properties, the combined branding potential of a single AOL portal that seamlessly integrates these properties would provide compelling value to the consumer. Despite their collective aversion to relinquishing control over the consumer, the carriers may elect to jointly co-brand and market AOL's portal. Similar reasoning holds for Yahoo and its Broadcast.com holding.

19.2.5. Other Short-Term Trends

ASR Vendors Evolve into Speech Gateway Vendors. While speech gateway vendors have always had close ties to the ASR industry, it seems as though the ASR vendors themselves are increasingly involved in the deployment of complete systems. Instead of residing on custom cards that provide a slow and expensive migration path, gateway components are being implemented on general purpose CPUs. Apart from the interface cards, all aspects of the speech gateway are realizable as server software: ASR, TTS, VoiceXML, Audio. For example, the Nuance ASR engine is widely employed in gateway systems created by external vendors, e.g. Tellme, BeVocal, Voice Genie, Motorola MIX, etc. However, Nuance itself offers its own Voice Web Server and even an adjunct browser-layer called Voyager, and it recently introduced its Vocalizer TTS engine to complete the lineup. This software expansion is not an isolated instance-SpeechWorks recently announced several products that similarly fill out its product line: the beta

version of its VoiceXML speech browser based on an open source interpreter of its creation[243], and it acquired Eloquent to expand its existing Speechify TTS offering.

The ASR industry's response may be closely tied to its fears of commoditization in a Voice Web world. With the dialog, synthesis, and grammar formats being standardized by the W3C and supported by the all segments of the voice industry, the only metrics along which the ASR vendors could differentiate themselves is performance and price. If the growth of the Voice Web further encourages programmers to eschew proprietary dialog components in favor of truly portable components, then the ASR players have no choice but to start vertically integrating along the platform chain.

19.3. Long-Term Trends

19.3.1. Emergence of Multiservice Providers

[243] "The Open VXI VoiceXML interpreter is a portable open source library that interprets the VoiceXML dialog markup language." See http://www.speech.cs.cmu.edu/openvxi/.

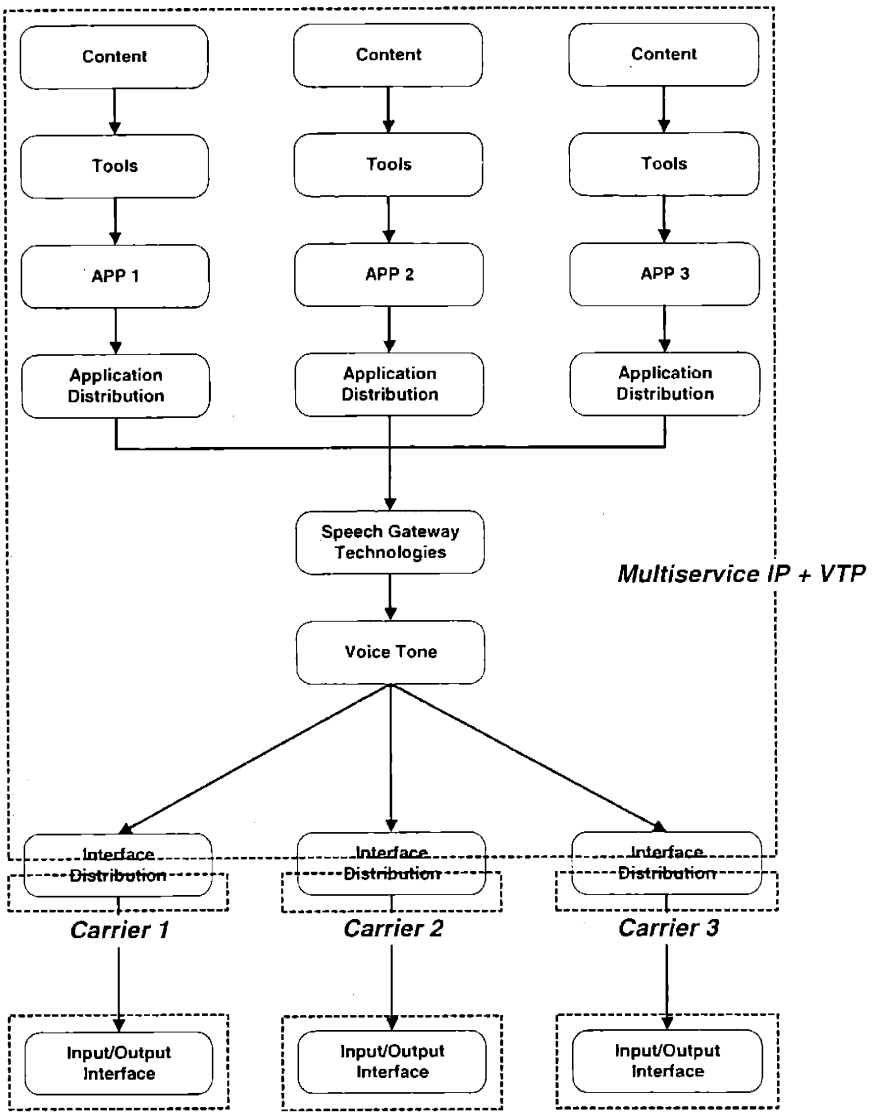

Figure 67. Multiservice Providers

As already alluded to in the previous section, Internet Telephony Service Providers and Content Delivery Network Operators share a common goal: building private networks to support real-time services over the Internet Protocol. Just as the speech gateway has evolved from a hodgepodge of custom hardware to server software running on general purpose computers, so too will formerly stand-alone services be considered as merely specific applications of a more basic packet layer, typically IP or ATM. CDNs and ITSPS stand on opposite side of the speech gateway, and both should be eager to incorporate that functionality to derive yet another revenue source from the basic IP service. As application providers increasingly turn to single source solutions for a variety of network services, it will only be a matter of time until the voice tone service itself is just another selection on the multiservice menu. Perhaps the only

market segment that cannot be integrated is the cellular service itself, at which point these multiservice monoliths have already been acquired by or have supplanted the traditional carriers.

19.3.2. Evolution of the End User Device

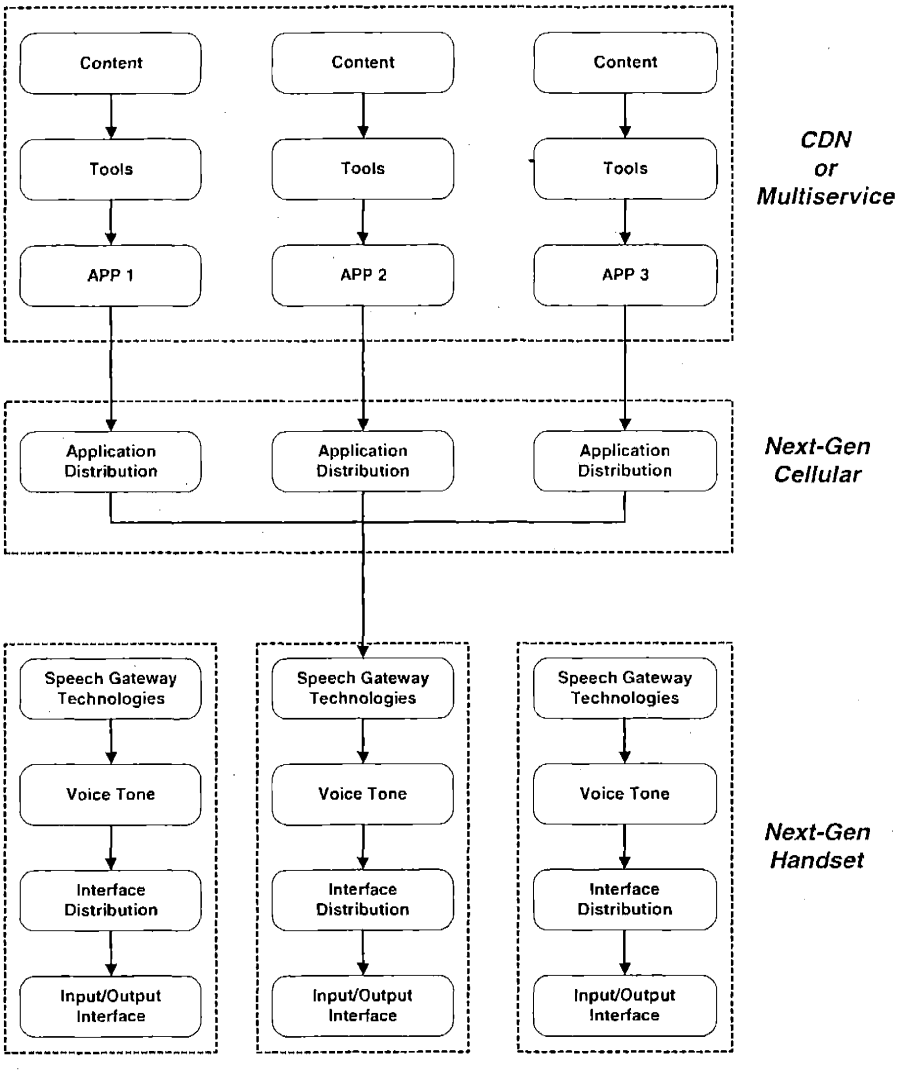

Figure 68. Next-Generation Handsets

The above discussion may be accelerated or side-stepped altogether if the cellular handset itself incorporates elements of the voice tone network. Specifically, the incorporation of the speech gateway would restore order to the client-server universe: VoiceXML interpretation, along with WML and any other desired XML language, would occur in the handset. This would certainly require a considerable upgrade in the processing power of a single phone, but then the WML browser so commonly found on

new handsets was at one point similarly inconceivable. The device would not be limited to 8-kHz telephony provided that the cellular network is prepared to offer wideband services, as opposed to further subdividing the spectrum into fixed lots to support more telephony users from a single cell.

At this point, the telephony service itself is hopefully just a service built on top of a more general QoS packet network, so it would be difficult for the next generation cellular providers to block access to the Voice Web and other content peer networks.

19.4. Recommendations to W3C and VoiceXML Forum

The Application-Speech Gateway Standards section outlined a number of language, interface, and domain name issues that are required for the establishment of the Voice Web, the most important of which are:

- Finalize dialog, grammar, and synthesis language standards: portability requires adherence to standards
- Support for streaming audio: branded content providers must not be excluded
- Flexible scoping levels across multiple documents: grammar management should not be an all or nothing proposition
- Universal Browser interface: navigation, audio, URI entry, bookmarking: no single site should be responsible for implementing the browser
- Vocal DNS: can't have a Web without the means to change addresses

The current state of the industry is firmly rooted in the vertical portal model, and the only sources to which a programmer or application provider can turn are biased toward a particular platform. There is absolutely no assurance that the VoiceXML Web will materialize, so branded content and application providers are naturally reluctant to embark on an extensive voice development project.

While the W3C must work with all due speed to finalize these standards, hopefully well in advance of the extremely conservative dates set by the Voice Browser Subgroup, the VoiceXML Forum itself can take some positive steps towards to realizing the Voice Web vision:

Application Testbed. The Voice Web movement suffers from a problem not unlike that of introducing a new computer: who will buy a computer without software, but then who will make software without

computer buyers-something is needed to jumpstart or bookstrap (bootstrap?) the market. Extending this logic voice space, the VoiceXML Forum can play a role in seeding the Voice Web with its first few seeds of content and ensure that they are well provided for as they incubate and evolve.

Most standards bodies would immediately reject any idea of subsidizing application development, particularly since some of the member companies in the Forum are actively pursuing this hosting market. The key difference is that audio content providers are extremely reluctant to get into the VoiceXML process itself, preferring instead to license the content to another organization willing to pick up the tab or simply forgo the opportunity entirely. Even though development sites already exist, they are always attached to a particular platform or voice tone provider, offering inconsistent advice and functionality from one provider to another.

As a proof case that the Voice Web can indeed be supported, the Forum can select a few marquee content providers for whom to develop and maintain W3C-compliant sites, all of which are accessible through a content delivery network. So instead of perpetuating the "non-model" of content licensing, this forces the service providers to step up to the challenge of accessing ad hoc information from outside the voice tone premises. By adamantly adhering to W3C recommendations, this puts pressure on the speech gateway vendors to adopt standards ahead of schedule. It could even be offered as a challenge or race for the industry at large, with the winner being the first voice tone facility to access the applications without modification or caching of any kind.

Community Building. While the VoiceXML community is deeply indebted to those pioneering portals that offer free tools and services to the development community, the Forum needs to step in and take responsibility for creating an arena for submitting best practices code, exchanging ideas, suggesting changes, with the overriding constraint that all code must be W3C-compliant and therefore portable. So instead of using Tellme's hidden intrinsic grammars or BeVocal's proprietary SpeechObjects, the development community would be submitting open source ECMAScript, VoiceXML subdialogs, etc. The Forum has already made steps to begin creating this community of people and resource: the electronic magazine VoiceXML Review, the formation of a VoiceXML Users Group, etc. The natural next step is to provide an electronic forum for exchanging ideas without regard for platform idiosyncrasies.

Chapter 20. CONCLUSIONS

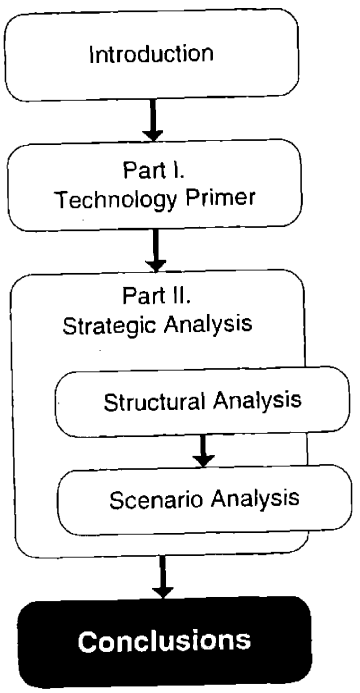

Figure 69. Overview of Paper (Currently on Conclusions)

"There's nothing on it worthwhile, and we're not going to watch it in this household."

- Philo T. Farnsworth's estimation of his own invention: the television

Given the frequency with which the term Voice Web is applied, one would assume that it is an unavoidable result or even already present reality . In the introductory chapter of this paper, the defining characteristics of the Voice Web were concisely enumerated: only one Voice Web; standard protocol and document formats; fair, consistent, and intuitive access to a Voice Web sites; and significant reuse of HTML Web resources. Throughout this paper, the reader has been presented with a variety of both enablers and barriers to achieving the Voice Web. This concluding discussion synthesizes the preceding analysis into a cogent argument **against** the viability of the Voice Web, but sounds a positive note for the role of speech input in next-generation, multimodal and embedded applications.

20.1. Barriers to the Voice Web

To be sure, the creation and rapid commercialization of VoiceXML signals a sea change in the voice industry. By avoiding the parochial mistakes of traditional ASR languages and embracing the client-server model of XML publishing, VoiceXML is the first and only standard that meets the criteria for delivering Voice Web-type applications: it is driven by an open standards process, supports dynamic content and callflow, and uses same hardware and communications resources as the HTML Web. Even in those cases where VoiceXML failed to establish appropriate and necessary standards (e.g. grammar and synthesis markup languages), the W3C has moved quickly to address these glaring omissions. Even a cursory review of gateway vendors aptly demonstrates that VoiceXML will fast become the *lingua franca* of the speech community, regardless of whether it is used for the traditional enterprise or content publishing industries. Perhaps most importantly, the infrastructure required to support VoiceXML services can be entirely assumed by the service provider community, allowing consumers to use regular telephones to access a wealth of audio content already on the Internet.

However, an effective dialog language alone does not a Voice Web make. In other words, the sustained viability of VoiceXML as a speech markup language is not synonymous with the creation of a Voice Web. As one might expect, the reasons for the Voice Web's market failure are rooted in a variety of factors:

1. For *inward-looking* companies that are simply looking to improve customer service or replace live agents, a VoiceXML-enabled speech gateway architecture may very well prove remarkably cost-effective on the bottom line; however, for *forward-looking* (e.g. content publishing) companies, the speech gateway presents an unacceptably high operational and cost burden on the existing HTML arhictecture. Considering the high per-port costs for hardware - interfaces, ASR/TTS/Audio engines, VoiceXML interpreters, and caching servers – as well as the significant recurring costs for telephony, one would be hard-pressed to find a convincing business model that can recover these expenses. So in the absence of an organization willing to assume the cost of gateway ownership and operation, publishers – the lifeblood of the Voice Web community - will simply forgo VoiceXML document creation altogether (with the possible exception of a highly-branded portal that can command a hefty monthly fee).

2. Given the natural economies of scale, the cellular carriers are the most likely candidates for assuming the role of VSP. Carriers are already experienced with operating telecommunications equipment and networks, and they are constantly looking for new value-added services to drive to their customer base. However, this reliance on the carriers ultimately forestalls the development

of the Voice Web for two reasons. First, since the VoiceXML interpreter resides entirely within the carrier's network, it can determine completely the universe of reachable sites (just like AOL's network), particularly since the availability of the DNS on a Voice Web service is still unclear. Second, the carrier can simply rely on per-minute pricing to discourage users from calling a competitor's portal. Since there is no such thing as a toll-free call in the cellular world and the consumer is loathe to squander her monthly allotment of minutes on information and entertainment services, the consumer will simply opt for the "house brand" offered at reduced rates by the carrier directly.

3. Even if a VoiceXML destination can be made available at low cost to consumers, the relevant question is whether the SIT-based Voice Web provides a compelling user experience. While it could be argued that ad hoc access to information services is a convincing feature, the issue of audio fidelity is another matter. Given that stereo-quality, branded audio is available for free both on the computer and on the radio, consumers may be reluctant to settle for the low-fidelity version over their cellular handsets. Unless consumers are uninterested in high-quality audio applications from their audio-only voice service, the low-fidelity rigor of the PSTN detracts from the Voice Web's utility and discourages the participation of branded audio publishing companies.

4. While VoiceXML does make it easier to create a speech application, it doesn't necessarily follow that VoiceXML reduces the difficulty of creating a good speech application. Whether as a result of our genetic makeup or years of convention, human beings are visual creatures: whereas we excel at visual parallel processing, we are incapable of listening to multiple audio sources and we cannot remember a long list of spoken items. While the availability of tools and best practices code will invariably raise of the general level of speech programming maturity, the author is comfortable in stating that the audio-only interface is entirely unnatural for browsing activities, which most certainly argues against the viability of the Voice Web. Good standalone speech applications are already difficult to implement, so a browsable, ad hoc universe of VoiceXML sites may simply be untenable from a human factors perspective.

5. Particularly damning is the absence of an appropriate DNS-type service for the Voice Web. It would be unreasonable to expect that users would take pleasure in spelling out each character of a URL any more than they enjoy using the telephone keypad enter URL into a WML browser, let alone the challenge of recognizing individual letters in a noisy environment. One would also prefer to avoid creating a parallel DNS system that prevents homonyms, similar-sounding words,

etc. If the universe of reachable content is defined by the currently active grammar alone, then this is nothing more than a traditional portal service and certainly not a content web.

It may well be the height of hubris to suggest that the much-heralded Voice Web will fail to materialize, but this conclusion is consistent with both the preceding analysis and the particular definition of the Voice Web. By relaxing the definition to allow for a multiplicity of incompatible portals, the Voice Web already exists; but by the same reasoning AOL is an integral part of the open HTML web. If the term "web" is to have any meaning, it must be applied with equal rigor to all application spaces, including the Voice Web.

20.2. Multimodal Web

In lieu of a true Voice Web, VoiceXML will likely initiate a period of growth in the traditional IVR enterprise market and perhaps the portal market as well. In spite of the anticipated market failure of the Voice Web, speech recognition will play an enduring role in enabling another vision: the Mobile Multimodal Web.

While the evolution of the handset may eventually erode the sustainability of the VSP market, it creates the opportunity for more robust multimodal applications. The availability of a screen for navigation, either as a primary or secondary interface, simply eliminates many of the previous human factors concerns: faster navigation through long lists, text-based input of URLs requires no modification to the DNS, simple creation of bookmarks on a personal browser, reduced reliance on unpleasant TTS and costly studio recording, familiar multimodal application design for HTML programmers, etc. Further, the device itself can utilize next-generation data networks to provide a more compelling user experience, including high-fidelity audio (and possibly video) and global positioning data.

From a cost perspective, the evolved handset essentially distributes the cost of the speech gateway directly to the users, so multimodal publication truly does not require any resources above and beyond traditional HTML publication. This serves the interests of the device manufacturers in that they must constantly evolve the product line to court carriers, and carriers are certainly looking to expand their revenues through the addition of data services. Interestingly, the carriers' embrace of multimodal devices ultimately removes its leverage in controlling the "walled garden" – they will be relegated to the position of commodity bit conveyor – but the competitive cellular market and the draw of additional minutes will secure the emergence of the Multimodal Web.

20.3. Embedded Speech Applications

In addition to the Multimodal Web, there will be parallel movement to incorporate limited forms of ASR technology into consumer devices, many of which may never be connected to any network. Instead of ASR that focuses on large and dynamic grammars, these embedded technologies support much more conservative tasks, e.g. recognizing the three words "stop," "pause," and "next" on a portable music player. In these cases, the goal is not to enable ad hoc browsing of VoiceXML sites, but to offer speech input as an integrated part of a device or appliance, such as toys, video game consoles, in-dash automotive system, etc. The emergence of speech as a natural part of the information ecosystem complements and even enables the multimodal vision of the mobile web.

20.4. Closing Remarks

Given the near ubiquity of traditional telephony and the convenience of a speech interface, there are many applications that naturally lend themselves to the SIT architecture; purchasing lottery tickets, remote track betting, live agent replacement, customer relationship management, etc. Most companies, including large enterprises, have yet to embrace speech applications, so the potential for traditional sales will likely remain for quite some time. In addition, the rapid commercialization of VoiceXML signals the continued strength of cooperation in the voice industry. However, these positive factors are simply insufficient to the task of creating and sustaining a Voice Web. While the author envisions an increasingly important role for speech recognition in the development of the Web and portable communications devices, it will not manifest itself as a Voice Web.