Pervasive Speech API Demo

Stefan Radomski
Technische Universität Darmstadt
Telecooperation
Darmstadt, Germany
sradomski@tk.informatik.tu-darmstadt.de

Dirk Schnelle-Walka
Technische Universität Darmstadt
Telecooperation
Darmstadt, Germany
dirk@tk.informatik.tu-darmstadt.de

***Abstract*—Current approaches for voice based interaction do not meet the special requirements of pervasive environments. While there is an increasing trend towards distributed systems, the concept of the classical client/server paradigm still prevails. We describe a framework wherein functional components are orchestrated dynamically, taking into account the users context and the changing availability and suitability of services in pervasive environments.**

***Keywords*-voice user interface; modality speech; pervasive environments; ubiquitous computing;**

I. Introduction

Pervasive environments pose unique challenges for human computer interaction that have not been met satisfyingly by current implementations. Major challenges arise from the fact that devices, available in these environments may not have an explicit display or physical interaction panel, e.g. a keyboard. The modality of speech offers a natural way to overcome these challenges. A voice user interface can enable a casual interaction, not requiring a user to pick up such explicit interaction devices. Furthermore, compared to tangible UIs, the modality of voice leaves a users hands and eyes free to perform other tasks or augment tangible UIs.

The pervasive speech API can be used to facilitate the development of multimodal applications where the user is on the move.

A. Description of the Demo

Within the demo, the output of a speech synthesizer or regular audio is directed to the output device most suitable for the current context of a user as determined, e.g. by her location. Imagine a user working at her desktop PC, listening to an audio book. As she leaves her working place, the audio output is redirected to a wearable headset or speakers in another room, following the location of the user. When she returns to her desktop, the audio output is continued on the speakers of her desktop PC.

The current implementation employs a wearable ultra mobile PC connected to an infrared receiver and a headset. Small infrared (IR) tags are placed within an environment to determine the users location and redirect the audio streams accordingly. Alternatively, a stationary setup with a mock-up of a roaming user represented by the IR receiver on a moving platform, can be deployed.

II. Technical Background

The available services in a smart environment (e.g. microphones as audio sinks and speakers as sources) register their capabilities and additional meta information within a directory, responsible for managing services in a set of given domains. A client can now query any such directory for services matching a given set of meta information or offering a set of given capabilities.

The meta-information of a service is encoded as an arbitrary tree of nested key-value pairs. The capabilities represent propositional logic formulas in disjunctive normal form, identifying valid parameter combinations in the space of all parameter combinations. A set of contractors can initiate contractual relations, if there is a non-empty intersection of their capabilities (i.e. there exists a common, true valuation for their formulas) and if they share a common domain.

Neither the parameters names for the capabilities, nor the available keys in the meta information are fixed, allowing and requiring application and service developers to identify and agree upon suitable taxonomies and parameters.

For the demo with audio sinks and sources, the parameters describing their capabilities include typical audio format settings such as the bit-depth, the sample rate, the number of available channels and the encoding. The meta information includes the name of the class implementing the service and its location in WGS84 coordinates [1].

A. Application Developers View

We chose the metaphor of *contract negotiations* to provide these concepts to application and service developers. With this metaphor, *offers* from *contractors* and dynamic offers as selected by *negotiators* at runtime can be combined into a set of common contract.

The process of using services in such an environment starts by defining a contract and adding the capabilities of queried services as offers from other contractors. The contract ensures that the capabilities of all contractors have a

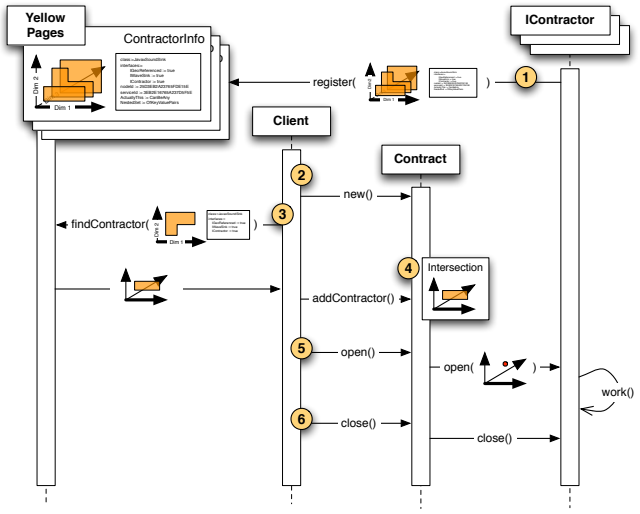

Figure 1. Process of contract negotiation

non-empty intersection, as the set of valid parameter combinations everyone can fulfill. In addition to adding offers from contractors directly, the user can select negotiators to be added to the contract. These negotiators are responsible for finding and maintaining contractual relations to the "best" service available. The requesting client does not necessarily have to take part as a contractor itself, allowing the initiation of contractual relations solely between remote contractors.

Each contractor can define a set of named presets for the parameters and a default as its preferred combination. When contractors are added to a contract, the framework tries to preserve the presets. An application developer can now choose one of the presets or let the framework determine an actual parameter combination from the intersection of all offers. Alternatively, the application developer can fix parameters within the contract to desired values. This will fail if the fixed value is not within the capabilities of all currently added contractors and prohibit the addition of unable contractors.

B. Modeling the Demo

For the demo, a set of sound sinks from different nodes, along with their locations are registered as contractors at a directory called *YellowPages*. The application sets up a contract, employing a single *ProximityNegotiator* to continuously search for the closest contractor offering audio output in a format as defined by the contracts parameters. The reference location of the negotiator is set by the IR receiver on the ultra mobile PC. Whenever a new IR tag is discovered, the tags location, as known a-priori, is provided to the proximity negotiator to reexamine the "best" suitable contractor and renegotiate the contract with the new service.

We wrote a small output stream class to provide the participating services in the contract with data. The output stream class is suitable to be used with our JSAPI2 implementation [2], allowing a seamless integration for implementations of Automatic Speech Recognition and Text-to-Speech system employing the JSAPI2 standard. By using this standardized API, new developers do not need to learn new concepts and can apply their previous knowledge.

To provide the functionality embodied in our contract metaphor, we use the Mundo ubiquitous computing framework [3], which offers a publish/subscribe paradigm and remote procedure calls on a wide variety of platforms.

III. Future Work

In order to realize our goal of a user tracking voice interface, we still need to integrate a speech recognition system. While our system would allow us to consume audio data from a stationary microphone close to the user, word recognitions rates would still not be sufficient to provide a convincing user experience. Headsets, on the other hand, inhibit acceptance as they involve a certain degree of preparation by the user and proved to be socially awkward. We are, therefore, about to conduct research with beam-forming microphone arrays [4][5] and sewn-in, miniaturized lavalier microphones.

IV. Conclusion

We described and implemented a conceptualization for service discovery and composition in pervasive environments based on the metaphor of contracts between participating services. Encoding the capabilities and information of these contractors as propositional logic formulas and nested key-values pairs respectively allows us to offer a framework to dynamically construct and adapt service compositions for multimodal user interfaces.

We employed the framework to implement part of a tracking Voice User interface, wherein an audio stream follows a roaming user in a pervasive environment.

References

[1] USA Department of Defense, *World Geodetic System (WGS)*, 1994.

[2] Conversay, *JSAPI 2.0 final release v2.0.6*, http://jcp.org/en/jsr/detail?id=113, Feb. 2009.

[3] E. Aitenbichler, J. Kangasharju, and M. Mühlhäuser, *MundoCore: A Light-weight Infrastructure for Pervasive Computing, Pervasive and Mobile Computing*, 2007, (332–361).

[4] P. Aarabi and S. Guangji, *Phase-based dual-microphone robust speech enhancement, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 2004, (Vol. 34 1763–1773).

[5] P. Aarabi, *The Fusion of Distributed Microphone Arrays for Sound Localization, EURASIP Journal on Applied Signal Processing*, 2003, (338–347).