GAIA: Common Framework for the Development of Speech Translation Technologies

Javier Pérez and Antonio Bonafonte

Department of Signal Theory and Communication
TALP Research Center
Technical University of Catalonia (UPC), Barcelona, Spain
{javierp,antonio}@gps.tsc.upc.edu

Abstract

We present here an open-source software platform for the integration of speech translation components. This tool is useful to integrate into a common framework different automatic speech recognition, spoken language translation and text-to-speech synthesis solutions, as demonstrated in the evaluation of the European LC-STAR project, and during the development of the national ALIADO project. Gaia operates with great flexibility, and it has been used to obtain the text and speech corpora needed when performing speech translation. The platform follows a modular distributed approach, with a specifically designed extensible network protocol handling the communication with the different modules. A well defined and publicly available API facilitates the integration of existing solutions into the architecture. Completely functional audio and text interfaces together with remote monitoring tools are provided.

1. Introduction

Multilingual human to human communication using machines is an area of investigation being actively developed. Different technologies are involved in the translation of speech, namely Automatic Speech Recognition (ASR), Spoken Language Translation (SLT) and Text-To-Speech synthesis (TTS). Several projects are currently under development or have recently been finished that make use of these technologies: ALIADO (2002), LC-STAR (Hartikainen et al., 2003), CHIL (2004), or TC-STAR (2004) to name a few. The development of the Gaia platform is directly embedded into LC-STAR.

LC-STAR was started in February 1st, 2002 and concluded on March, 2006, and focused on the creation of language resources for transferring Speech-to-Speech Translation components. The goal of the project was to improve human-to-human and person-machine communication in multilingual environments, aiming at the creation of the needed lexica and corpora for flexible vocabulary speech recognition, speech centered translation into selected languages and high-quality speech synthesis. Two parallel tracks addressed the different maturity of speech-to-speech translation components. The goal of the first track was to develop pronunciation lexica for each of the 12 languages, consisting of 50000 common word entries and 50000 proper names entries each. The second track included the development of corpora and language resources needed for speech-to-speech translation, and the creation of a baseline system. Among the different subtasks defined in the later track, the Gaia platform was proposed as a demonstration system to show the language transfer. The rest of this paper is devoted to the description of the platform architecture.

2. Architecture

The demonstrator platform, Gaia, is a distributed server which can offer translation in the three research languages covering a tourist domain defined in the project. The platform structure follows a modular design, where different parts can be launched in separate machines.

Two different paradigms could be followed for achieving distribution. The whole platform including the different client interfaces, the recognition, translation and synthesis engines, and the platform kernel could be implemented as objects or components of a single entity. In this case, several distributed computing models exist that permit components to be spread across multiple computers: CORBA (Common Object Request Broker Architecture)(Obj, 1998) or OOA (Open Agent Architecture) (Cheyer and Martin, 2001) among others. This is the paradigm followed in the related CHIL (2004) project, aimed at the exploration and creation of environments in which computers are used in human to human communication. On the other hand, a client/server approach could be followed where each part is conceived as an individual entity, with the ability (or requirement) to run independently of the platform. This is the approach followed in the Galaxy (Senef et al., 1998) project at MIT or in the concluded national Spanish project BASURDE (Bonafonte et al., 2000) among others.

The development of the Gaia platform follows the later paradigm, since our interest is to integrate existing solutions into a common framework and the client/server approach facilitates this process. Using a distributed computing paradigm increases unnecessarily the complexity of the system and penalizes the integration of any existing technology, since it requires heavy modifications of the source code. Standard sockets and a predefined network protocol are used to interconnect the different modules of the system. Figure 1 shows the different modules of the platform and its distribution across a network environment. The speech processing tasks (recognition, translation and synthesis) are implemented in independent servers, communicating with the platform kernel via standard network sockets. Distribution is mandatory since each task can be computationally demanding. Moreover, it allows the different partners to implement their particular solutions in different platforms if desired, achieving a high grade of flexibility. The kernel of the platform communicates with different

types of servers:

- Terminal servers: they collect the input of the user and provide the output. Three dual terminal servers have being developed: i) telephone terminal, to interact using the telephone through Dialogic cards; ii) speech console terminal, to interact using speech through and IP connection and iii) text console terminal which is mainly used to test the translation engine.

  In LC-STAR, the demonstration is based on the telephone terminal: two users can communicate using different languages, through a translation service provided by Gaia through the telephone.

- Speech Technology servers. Gaia is prepared to compare different technologies. Therefore, more than one server for a given technology can exist.

During the LC-STAR project, the following technology servers were integrated:

- Automatic Speech Recognition (ASR), provided by UPC, see Bonafonte et al. (1998b) or Mariño et al. (2000).

- Spoken Language Translation (SLT), using both the technology provided by RWTH (Och and Ney, 2002) and by UPC. The former included the advances achieved during the LC-STAR project.

- Text to Speech Synthesis (TTS) provided by UPC (Bonafonte et al., 1998a) for Spanish and Catalan, and by the Festival project (Black and Taylor, 1997) for English.

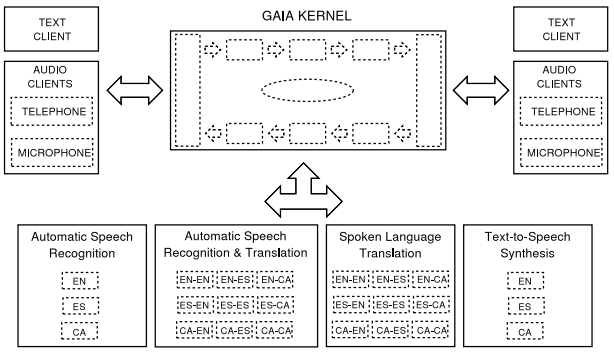

Figure 1: *Schematic diagram of the complete Gaia platform: text and audio clients, and ASR, ASR with integrated translation, SLT and TTS servers for the available languages.*

The Gaia kernel handles the communication among the terminals (platform clients) and the different speech technology servers. The kernel keeps a database with the network location (IP address and port) of the different ASR, SLT and TTS servers, and establishes the appropriate connections based on the language, preferences and characteristics of the client. Several options are supported to select the pair of languages for the source and target terminal: either the source client sets both, or the destination client selects its language. Predefined pairs of languages are also available during the configuration stage. Depending on whether the terminal servers use audio or text, the platform uses the speech recognizers and synthesizes accordingly. An example of this flexibility is shown in fig. 2, where a client using Catalan connects to the platform using and audio terminal (handset), and connects to the destination client which uses English and a text interface. In this case, no synthesizer is needed after the source to target translation stage, and no recognition server is needed for the target client.

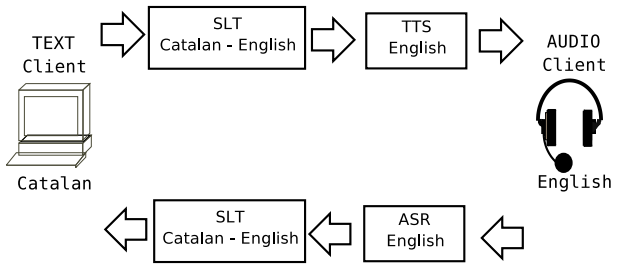

Figure 2: *Schematic diagram of the case of an English audio client connecting to a Catalan text client.*

3. Speech Technology Modules

As seen in a previous section, Gaia follows the client/server approach to account for the distribution of tasks. We have implemented a network protocol to communicate with the different servers involved in the process. Due to space restrictions, we will only present here the generalities of the protocol, and the different issues that are accounted for. For a detailed description of the protocol and other aspects of the software implementation, the reader is referred to the technical documentation (Pérez, 2004). A common structure is shared by the different structures, incorporating commands to start, pause, resume and stop the operation of the server. An abort possibility is available to immediately stop the server in case some exceptional situation occurs (network failure, user disconnection, etc.).

3.1. Automatic Speech Recognition

Gaia can be configured to use several independent recognizers, and the protocol contemplates the possibility of a different grammar for each session; in its current implementation, the grammar used by the server can not be changed on a per turn basis. The connection to the server is not closed and reopened again each time there is a change of turn. Instead, it remains open for the whole session, making it possible to use different adaptive techniques in recognition. For instance, the models could be adapted to the speaker or the different background noise conditions.
Along with the recognized text, it is often necessary to have detailed information regarding the confidence of the text and the position of the word inside the speech stream. The protocol allows the recognizer server to return a special structure containing this information. The protocol incorporates several commands to set some timing parameters related to the proper ASR operation: maximum allowed initial silence, maximum allowed end silence, and a maximum utterance duration. Different situations require the speech

to be sampled with a particular frequency, hence it is desirable that the protocol allows for several speech stream parameters to be changed Among others, the sampling frequency and compression scheme (A-law, μ-law and linear PCM) can take different values at each turn.

3.2. Spoken Language Translation

Two different approaches to spoken language translation are contemplated by the Gaia platform. First, the platform implements the translation step in an engine (i.e. server) independent from that of the ASR. The current SLT server API has been kept quite simple, since the translation track of the LC-STAR project is under active development and no requirements other than to send and receive the original and translated text, respectively, have been set. Needless to say, the common set of commands shared by all the different platform interfaces is available (start, pause, resume and abort operation).
On the other hand, one of the implemented prototypes integrates the translation engine in the ASR. In this case, no specific SLT server is required, since the text received from the recognition server is already translated into the target language. As figure 1 shows, in case of integrating recognition and translation, an ASR server must be defined for each pair of languages. This is not necessary if using independent engines, since no knowledge about the target language is required in the recognition step.

3.3. Text-To-Speech Synthesis

We are currently using our own UPC synthesis engine and the Festival synthesis system. The existing implementation of the synthesizer interface only incorporates commands to configure the audio stream. Similar to the recognition interface, the compression schema (A-law, μ-law and linear PCM) and the sampling frequency can be set on a per turn basis. The common approach of using a standardized markup language such as SSML [1] can be used to set other options in the synthesis server. Standard marks exist to select which speaker to use for synthesizing, the rate of the generated speech and whether we want an special *pronunciation* mode to be used (e.g. spelling an acronym or saying a date or proper name). We adopted this approach since several existing TTS systems already include it, thus no modifications are required.

4. Usage scenarios

Gaia has been developed with flexibility as one of the main goals. Hence, multiple configurations of the platform can be selected at runtime, depending on the desired scenario. The complete configuration would be with two different clients, and an interface for the ASR, the SLT and the TTS for each direction of the call flow. But multiple variations are allowed, where some of the interfaces may or not be present. Imagine a scenario where two users want to use the platform to communicate to each other in a different language using the telephone. This requires the *complete* version of the platform to be active, since both clients have an audio interface, thus requiring the recognition and synthesis stages to be performed, and the text translation engine is necessary due to the different languages involved. For each client, the speech recognizer is used to obtain the transcription of what the user said, the translation would be obtained using a standalone text translation server, and then this text would be sent to the speech synthesizer in order to obtain the audio data for the destination user.
A slightly different scenario would occur if the translation engine and the speech recognizer were integrated. In this case, the output text obtained from the recognizer could be directly sent to the synthesizer module, since it would already be translated, and there would be no need to use a different SLT. A direct connection between the ASR interface and the TTS interface would be set by the kernel, and no SLT interface would be created.
It is also possible for each user to have a different interface to access the platform (e.g. one using a telephone client, and the other a text client). It is clear that in one direction the ASR module would not be necessary, whereas in the other the synthesis part can obviously be skipped. The client interface in the kernel assigned to the text client would send and receive data directly from the translation interfaces, while the interface connected to the audio client would send data to the recognizer, and receive its input from the TTS interface (this is the case previously shown in figure 2).
There exist several scenarios where only one client uses the platform. For instance, when testing a SLT module the client would send the text to be translated to the platform, and would directly receive the translated strings. A similar situation occurs when Gaia is used as a tool to record controlled dialogues. In this case, one of the clients would be the user speaking the utterances to be recorded, and the other client would be a program reading recorded prompts (i.e. audio files) and sending them to the platform (fig. 3). This offers many interesting possibilities. For instance, we could have a recording scenario where the user repeats exactly what is received from the platform. Another useful option would be to implement a question/answering system, where the questions are stored on disk, and the client answers recorded by the platform. Figure 3 illustrates this general scenario, which is related to the functionality of the Ada (Rodríguez and Moreno, 1998) tool developed at our university.

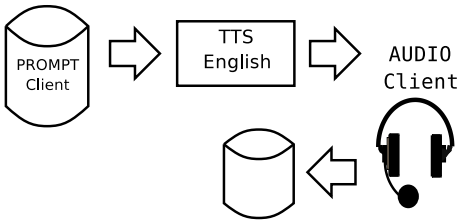

Figure 3: *Diagram of Gaia platform used as a prompt recorder.*

5. Conclusions

We have presented in this paper a software environment that can be useful when doing research in spoken language

[1] http://www.w3.org/TR/speech-synthesis/

technologies. The principal role of the Gaia platform is to connect different solutions for each of the technologies involved (automatic speech recognition, spoken language translation and text-to-speech synthesis). Considering the high computational requirements of each individual technology and the flexibility required when multiple partners collaborate providing a particular server Gaia (with the possibility of each of them working in a different platform), it is mandatory to minimize the processor load of the platform. Furthermore, there is need of distributed computing, since we do not desire to overload a single processor or machine. After careful review of several related projects, a client/server approach has been adopted and an extensible network protocol specifically designed. This facilitates the integration of existing independent servers into the Gaia framework. The existing UPC servers for ASR, SLT and TTS have been adapted and integrated into the platform structure. We have also implemented an intermediate layer in order to use the Festival Speech Synthesis System (Black and Taylor, 1997) with the Gaia platform.

Several demonstrations of speech translation have been performed (Banchs et al., 2006), using either the integrated recognition and translation engine, or the separated SLT solution. The statistical translation engine developed at RTWH (Och and Ney, 2002) was directly integrated into the text translation engine as a dynamic library. As mentioned earlier in this paper, the Spanish project ALIADO (ALIADO, 2002) has also adopted it as the demonstration platform. The speech corpus needed for the LC-STAR (Hartikainen et al., 2003) project has been obtained using this platform.

The Gaia platform has already been successfully used during the development of different stages in the LC-STAR project, as reported in (Banchs et al., 2006). It was used to obtain the speech corpora belonging to the travel domain needed for the project by selecting a special configuration, since no recognition, translation nor synthesis was required, only the recorded audio.

Portability among different platforms is guaranteed by using standard C++ for programming the different parts. The remote monitoring and configuration applets are programmed in Java and can be used from several Java-compatible browsers (e.g. Mozilla, Galeon or Opera). Since the software is provided together with full sources and technical documentation, developers will find it easy to adapt any related technology to Gaia.

6. Acknowledgements

This work has been partially sponsored by the European Union under grant IST-2001-32216 and the Spanish Government under grant TIC2002-04447-C02.

7. References

ALIADO, 2002. *ALIADO, Tecnologías del habla y el lenguaje para un asistente personal*. http://gps-tsc.upc.es/veu/aliado/main.html.

Rafael Banchs, Antonio Bonafonte, and Javier Pérez. 2006. Acceptance testing of a spoken language translation system. In *LREC'06, Genoa, Italy*, May.

A. Black and P. Taylor. 1997. Festival speech synthesis system: system documentation. Technical report, Human Communication Research Centre. Technical Report HCRC/TR-83.

Antonio Bonafonte, Ignasi Esquerra, Albert Febrer, José A. R. Fonollosa, and Francesc Vallverdú. 1998a. The UPC Text-to-Speech System for Spanish and Catalan. In *Proc. ICSLP 98, Sydney, Australia*, November.

Antonio Bonafonte, José B. Mariño, Albino Nogueiras, and José Adrián Rodríguez Fonollosa. 1998b. RAMSES: el sistema de reconocimiento del habla continua y gran vocabulario desarrollado por la UPC. In *VIII Jornadas de Telecom I+D (TELECOM I+D98)*, Madrid, Spain, October.

Antonio Bonafonte, Pablo Aibar, Núria Castell, Eduardo Lleida, José B. Mariño, Emilio Sanchis, and M. Inés Torres. 2000. Desarrollo de un sistema de diálogo oral en dominios restringidos. In *I Meeting on Language Engineering, Sevilla, Spain*, November.

Adam Cheyer and David Martin. 2001. The open agent architecture. *Journal of Autonomous Agents and Multi-Agent Systems*, 4(1):143–148, March.

CHIL, 2004. *CHIL, Computers in the Human Interaction Loop*. http://chil.server.de.

E. Hartikainen, G. Maltese, A. Moreno, S. Shammass, and U. Ziegenhai. 2003. Large lexica for speech-to-speech translation: From specification to creation. In *Eurospeech*. LC-STAR homepage: http://www.lc-star.com/.

José B. Mariño, A. Nogueiras, P. Pachès, and A. Bonafonte. 2000. The demiphone: an efficient contextual subword unit for continuous speech recognition. *Speech Communication*, 32(3):187–197, oct.

Object Management Group (OMG), 1998. *CORBA/IIOP 2.2 Specification*, February. Available online at http://www.omg.org/corba/corbiiop.html.

Franz J. Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *ACL02*, pages 295–302, Philadelphia, PA, July.

Javier Pérez, 2004. *Gaia technical report*. TALP Research Center. http://gps-tsc.upc.es/veu/personal/javierp/.

José Adrián Rodríguez and Asunción Moreno. 1998. Automatic database adquisition software for isdn pc cards and analogue boards. In *LREC'98, Granada, Spain*, May.

S. Senef, E. Hurley, R. Lau, C. Pao, P. Schmid, , and V. Zue. 1998. Galaxy-II: A reference architecture for conversational system development. In *Proc. ICSLP 98, Sydney, Australia*, November.

TC-STAR, 2004. *TC-STAR, Technology and Corpora for Speech to Speech Translation*. http://www.tc-star.org.