Collaborative Project

EU-BRIDGE

Bridges Across the Language Divide

Grant Agreement Number 287658

EB Client – MCloud Service Architecture

Version: 2.0 Date: 21/05/2014

Type of activity	RTD		**Work Package**	WP 7	
Due date	21/05/14		**Submission date**	21/05/14	
Main author(s)	Andrea Franzoso, PerVoice SpA				
Co-authors					
Reviewers					
Contributors					
Version(s)	V1.0	**Status**	Review	**Date**	
		Status	Final	**Date**	
Dissemination level			**Nature**	R / O	
Keywords	Service Architecture, Client, MCloud,				

P = Prototype; **R** = Report; **O** = Other; **D** = Demonstrator

Part of the Seventh Framework Programme

Funded by the EC – DG CONNECT

Executive Summary (Abstract)

The current document aims at providing the basic indications on the utilization of the EB Client, which allows to connect to the Service Architecture and subscribe to one or multiple input streams of different types of languages in order to get for instance the translations results.

Table of Contents

1. Introduction

The Service Architecture represents an efficient Distributed Architecture, which is characterized by message protocol hidden by the API, XML message format, communications based on TCP/IP sockets connections based and an asynchronous sending and processing of packets using queues and callback functions.

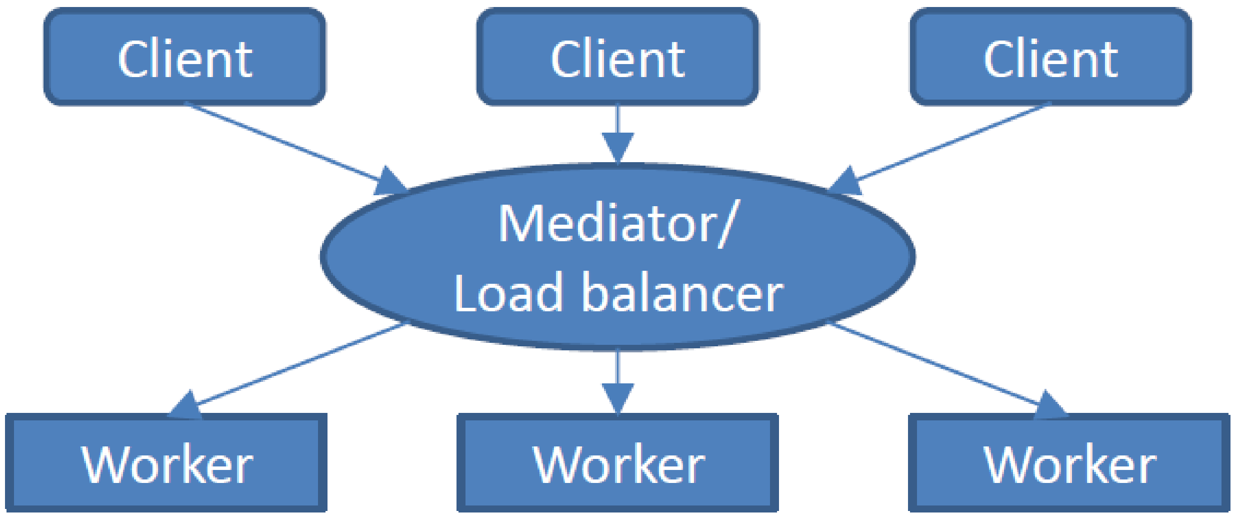
The EB Client is an executable file (.exe under Windows) which can be run in a command prompt under a Windows environment and in a shell under a UNIX environment.

2. Basic Scheme of the Infrastructure

The Service Architecture is based on a client - server configuration, where clients can connect and send media streams (text, audio, …), while the servers, that are called "Workers", can accepts one incoming service request per connection after registering with one or multiple services that are able to handle.

In the middle, a Mediator takes and processes the requests of each client and activates the specific worker, which was recognized as able to handle the desired services.

The basic scheme of the infrastructure is therefore as following:

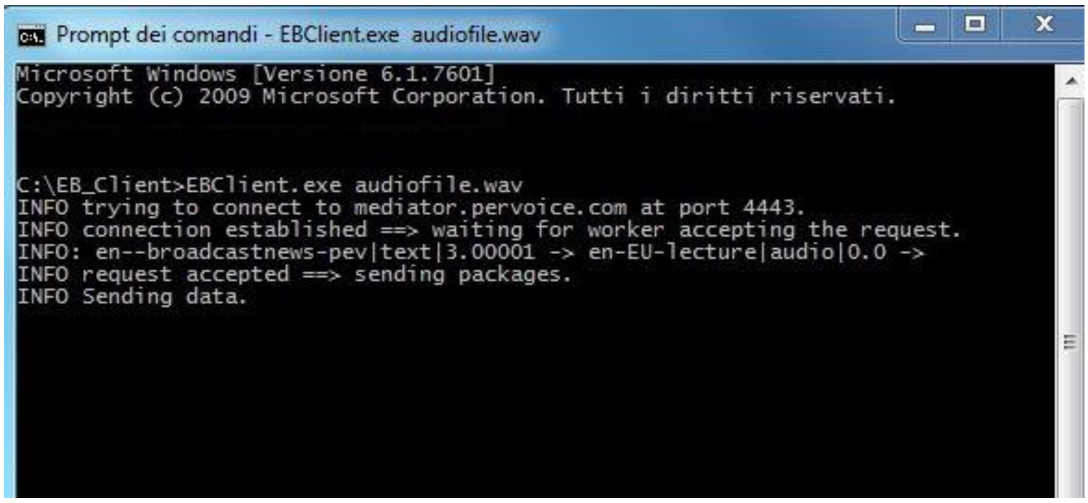


3. EB Client

The EB Client is an executable file that simulates the behavior of a generic client both in a Windows and in a UNIX environment. It can connect to the Service Architecture and requests the services the MCloud Library can handle.

Once in the correct directory in a Windows command prompt or in a UNIX shell, the EB Client can be basically run with the following syntax:

./name_ of_the_executable_file [OPTION]... <audio file>



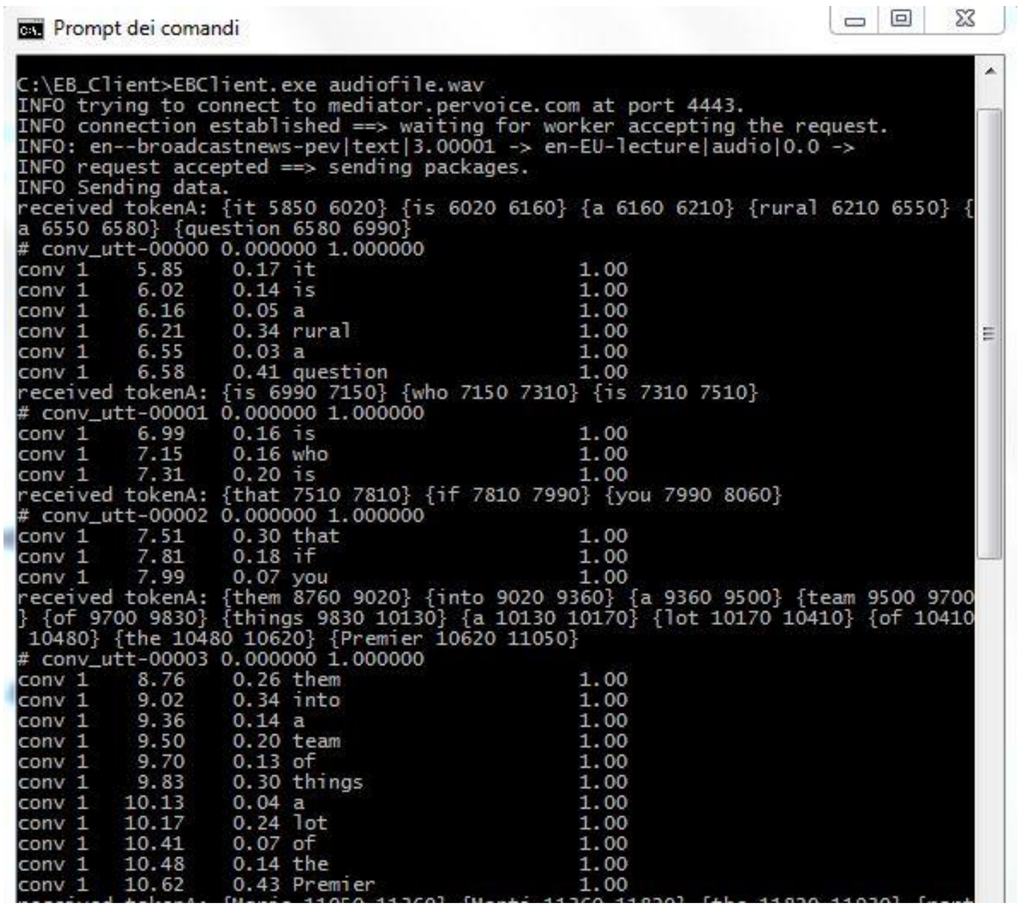
Prompt dei comandi - EBClient.exe audiofile.wav

```
Microsoft Windows [Versione 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Tutti i diritti riservati.


C:\EB_Client>EBClient.exe audiofile.wav
INFO trying to connect to mediator.pervoice.com at port 4443.
INFO connection established ==> waiting for worker accepting the request.
INFO: en--broadcastnews-pev|text|3.00001 -> en-EU-lecture|audio|0.0 ->
INFO request accepted ==> sending packages.
INFO Sending data.
```

Windows Command prompt showing how to run the client

After some seconds, client will receive a reply from the server and start the communication.



Prompt dei comandi

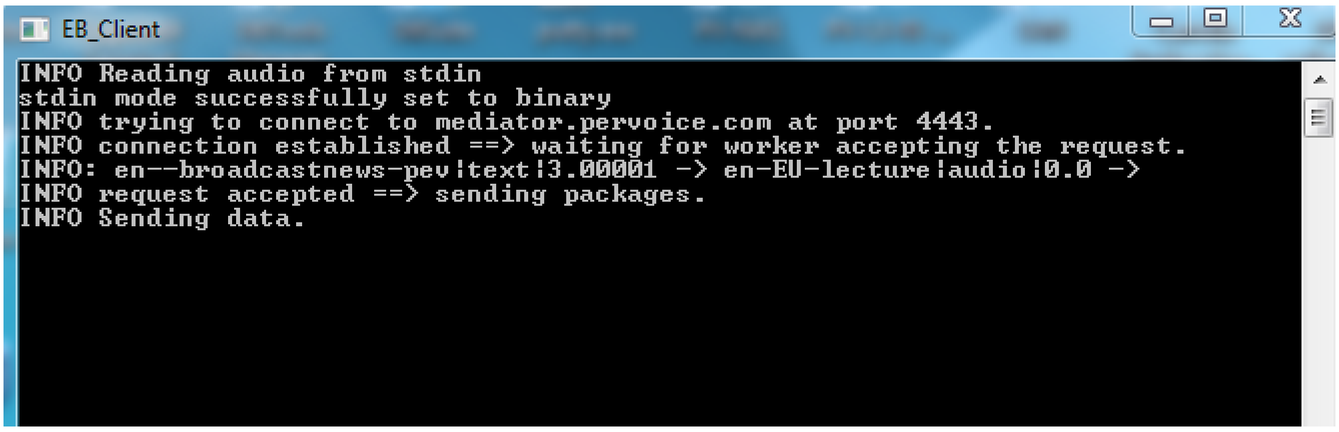
```
C:\EB_Client>EBClient.exe audiofile.wav
INFO trying to connect to mediator.pervoice.com at port 4443.
INFO connection established ==> waiting for worker accepting the request.
INFO: en--broadcastnews-pev|text|3.00001 -> en-EU-lecture|audio|0.0 ->
INFO request accepted ==> sending packages.
INFO Sending data.
received tokenA: {it 5850 6020} {is 6020 6160} {a 6160 6210} {rural 6210 6550} {
a 6550 6580} {question 6580 6990}
# conv_utt-00000 0.000000 1.000000
conv 1     5.85     0.17 it                         1.00
conv 1     6.02     0.14 is                         1.00
conv 1     6.16     0.05 a                          1.00
conv 1     6.21     0.34 rural                      1.00
conv 1     6.55     0.03 a                          1.00
conv 1     6.58     0.41 question                   1.00
received tokenA: {is 6990 7150} {who 7150 7310} {is 7310 7510}
# conv_utt-00001 0.000000 1.000000
conv 1     6.99     0.16 is                         1.00
conv 1     7.15     0.16 who                        1.00
conv 1     7.31     0.20 is                         1.00
received tokenA: {that 7510 7810} {if 7810 7990} {you 7990 8060}
# conv_utt-00002 0.000000 1.000000
conv 1     7.51     0.30 that                       1.00
conv 1     7.81     0.18 if                         1.00
conv 1     7.99     0.07 you                        1.00
received tokenA: {them 8760 9020} {into 9020 9360} {a 9360 9500} {team 9500 9700
} {of 9700 9830} {things 9830 10130} {a 10130 10170} {lot 10170 10410} {of 10410
 10480} {the 10480 10620} {Premier 10620 11050}
# conv_utt-00003 0.000000 1.000000
conv 1     8.76     0.26 them                       1.00
conv 1     9.02     0.34 into                       1.00
conv 1     9.36     0.14 a                          1.00
conv 1     9.50     0.20 team                       1.00
conv 1     9.70     0.13 of                         1.00
conv 1     9.83     0.30 things                     1.00
conv 1    10.13     0.04 a                          1.00
conv 1    10.17     0.24 lot                        1.00
conv 1    10.41     0.07 of                         1.00
conv 1    10.48     0.14 the                        1.00
conv 1    10.62     0.43 Premier                    1.00
```

Windows Command prompt showing the communication client - server

It also includes a list of options that can be used and that are described in the following paragraphs.

The audio file must be for the moment a *wav* file.

By double clicking on the exe file, the Windows command prompt will automatically open and the client will wait for some audio coming from stdin.



```
INFO Reading audio from stdin
stdin mode successfully set to binary
INFO trying to connect to mediator.pervoice.com at port 4443.
INFO connection established ==> waiting for worker accepting the request.
INFO: en--broadcastnews-pev|text|3.00001 -> en-EU-lecture|audio|0.0 ->
INFO request accepted ==> sending packages.
INFO Sending data.
```

Windows Command prompt showing the client sending data coming from stdin

4. Example of use with SoX

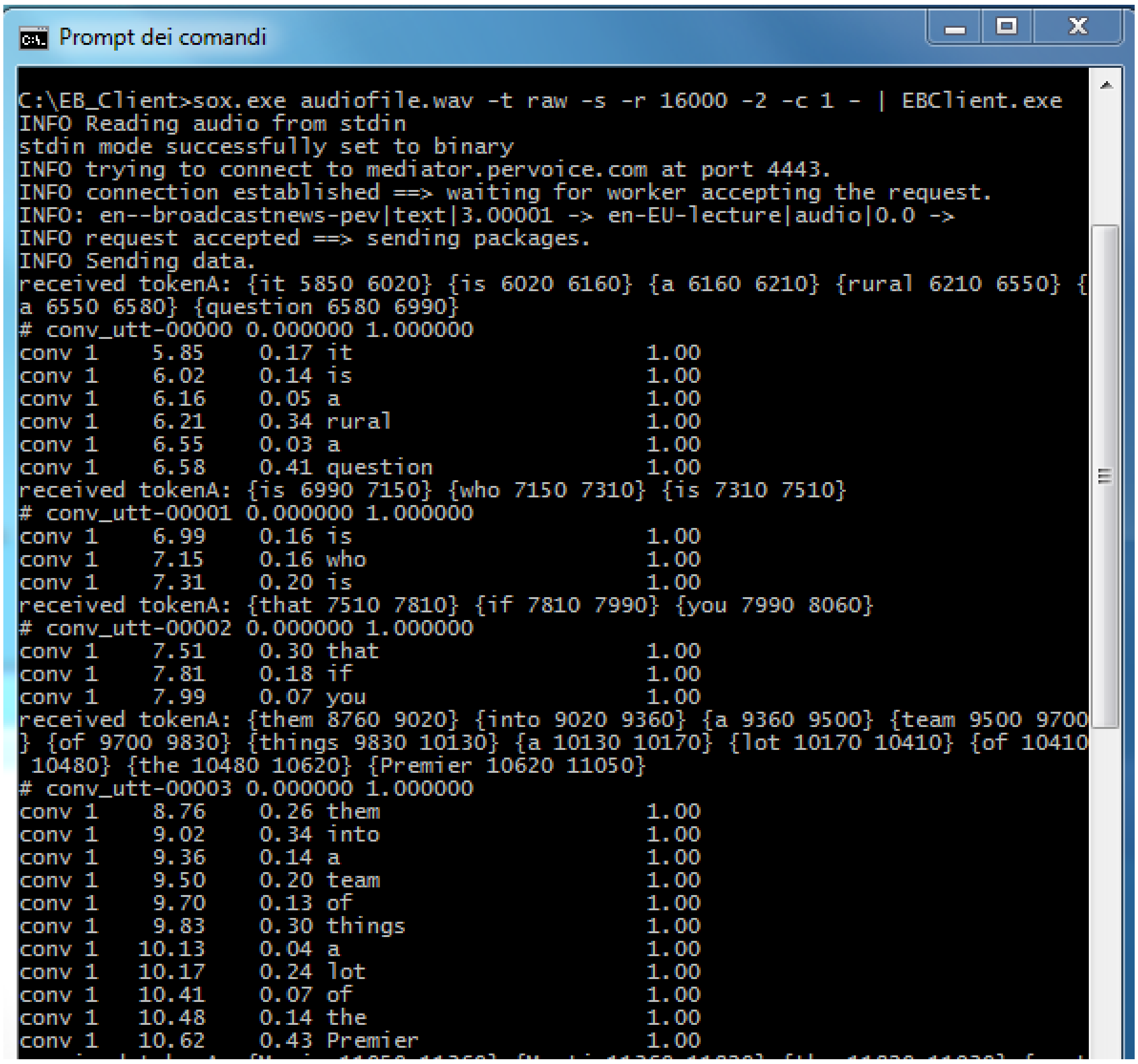
With a recent implementation, the audio file can be omitted and the EB Client will then consider as input an incoming stream to the shell.

This behavior can be easily simulated by means of the SoX software, which can create an input stream in STDIN. After installing SoX and its libraries, the specific syntax to run this type of simulation is the following:

sox.exe <audio file> -t raw -s -r 16000 -2 -c 1 - | *./name_of_the_executable_file*

The SoX part generated the input stream, which is accepted by the EB Client executable by means of a piping operation.

In the previous synopsis the SoX parameters is chosen so to have a 16 KHz bitrate and a single audio channel.



```
C:\EB_Client>sox.exe audiofile.wav -t raw -s -r 16000 -2 -c 1 - | EBClient.exe
INFO Reading audio from stdin
stdin mode successfully set to binary
INFO trying to connect to mediator.pervoice.com at port 4443.
INFO connection established ==> waiting for worker accepting the request.
INFO: en--broadcastnews-pev|text|3.00001 -> en-EU-lecture|audio|0.0 ->
INFO request accepted ==> sending packages.
INFO Sending data.
received tokenA: {it 5850 6020} {is 6020 6160} {a 6160 6210} {rural 6210 6550} {
a 6550 6580} {question 6580 6990}
# conv_utt-00000 0.000000 1.000000
conv 1    5.85    0.17 it                   1.00
conv 1    6.02    0.14 is                   1.00
conv 1    6.16    0.05 a                    1.00
conv 1    6.21    0.34 rural                1.00
conv 1    6.55    0.03 a                    1.00
conv 1    6.58    0.41 question             1.00
received tokenA: {is 6990 7150} {who 7150 7310} {is 7310 7510}
# conv_utt-00001 0.000000 1.000000
conv 1    6.99    0.16 is                   1.00
conv 1    7.15    0.16 who                  1.00
conv 1    7.31    0.20 is                   1.00
received tokenA: {that 7510 7810} {if 7810 7990} {you 7990 8060}
# conv_utt-00002 0.000000 1.000000
conv 1    7.51    0.30 that                 1.00
conv 1    7.81    0.18 if                   1.00
conv 1    7.99    0.07 you                  1.00
received tokenA: {them 8760 9020} {into 9020 9360} {a 9360 9500} {team 9500 9700
} {of 9700 9830} {things 9830 10130} {a 10130 10170} {lot 10170 10410} {of 10410
 10480} {the 10480 10620} {Premier 10620 11050}
# conv_utt-00003 0.000000 1.000000
conv 1    8.76    0.26 them                 1.00
conv 1    9.02    0.34 into                 1.00
conv 1    9.36    0.14 a                    1.00
conv 1    9.50    0.20 team                 1.00
conv 1    9.70    0.13 of                   1.00
conv 1    9.83    0.30 things               1.00
conv 1   10.13    0.04 a                    1.00
conv 1   10.17    0.24 lot                  1.00
conv 1   10.41    0.07 of                   1.00
conv 1   10.48    0.14 the                  1.00
conv 1   10.62    0.43 Premier              1.00
```

Windows Command prompt showing how to run the client in pipe with SoX

5. Available Options

The EB Client provides a list of useful option that can be used for instance to change the server and the port which to connect to, or to define the fingerprint.

In the following, a description of all the possible options is presented.

Option	**Description**
-s --serverHost=HOSTNAME	Hostname of the server where the Mediator is running
-p --serverPort=PORT	Port address at which the Mediator accepts the workers
-f --fingerprint=FPRINT	Language fingerprint
-i --inputType=FPRINT	Type of the desired results
-l --logging	Option to turn on/off the logging of submitted data
-w --writeCTM=FILE	This option writes the results into a NIST CTM file
-n --conv=ID	Conversation ID of the NIST CTM file
-x --plaintxt	This option provides the output in plain text
-C --codec=CODEC	Audio codec used to transmit and receive data to/from the Mediator Currently supported codecs are OPUS, SPEEX and FLAC.
-S --sampleRate=SRATE	Sample rate used to transmit and receive data to/from the Mediator
-B --bitRate=BRATE	Bit rate used to transmit and receive data to/from the Mediator
-	This option, if specified, allows to deal with input stream directly from STDIN. The client is therefore listening to accept the data
stdin	This option is an alternative of "-"; leaving the "–x" option blank leads to the same result, i.e. if the "–x" option is not specified, the client considers as input the data from STDIN
-h --help	This option shows the help

The options can be used simultaneously and have to be specified, otherwise the default values are chosen. These values are:

Option	**Default Value**
serverHost	mediator.pervoice.com
serverPort	4443
fingerprint	en-EU-lecture
inputType	text
logging	0 (if specified, default username "SA" and default password "SA")
conv	conv
plaintxt	dataCallback, otherwise the input stream is taken from STDIN
codec	RPCM (raw PCM)
sampleRate	16000 [Hz]
bitRate	32000 [bit/s]

6. Code

The code of the EB Client is in the exampleClientBidir.c file which can be compiled both in a Windows and in a Linux environment with the presence of the MCloud library.