

## Table of Contents

Overview of TTRemoteClient.exe .....	1
Initial Launch of TTRemoteClient.exe .....	1
Configuring a Client.....	2
Selecting a Client to Test.....	4
Notes on the different Client Types.....	5
Testing using the ASCII TTServer Interface .....	6
ASCII Radio Command List .....	8
Heading 3 .....	8

## Overview of TTRemoteClient.exe

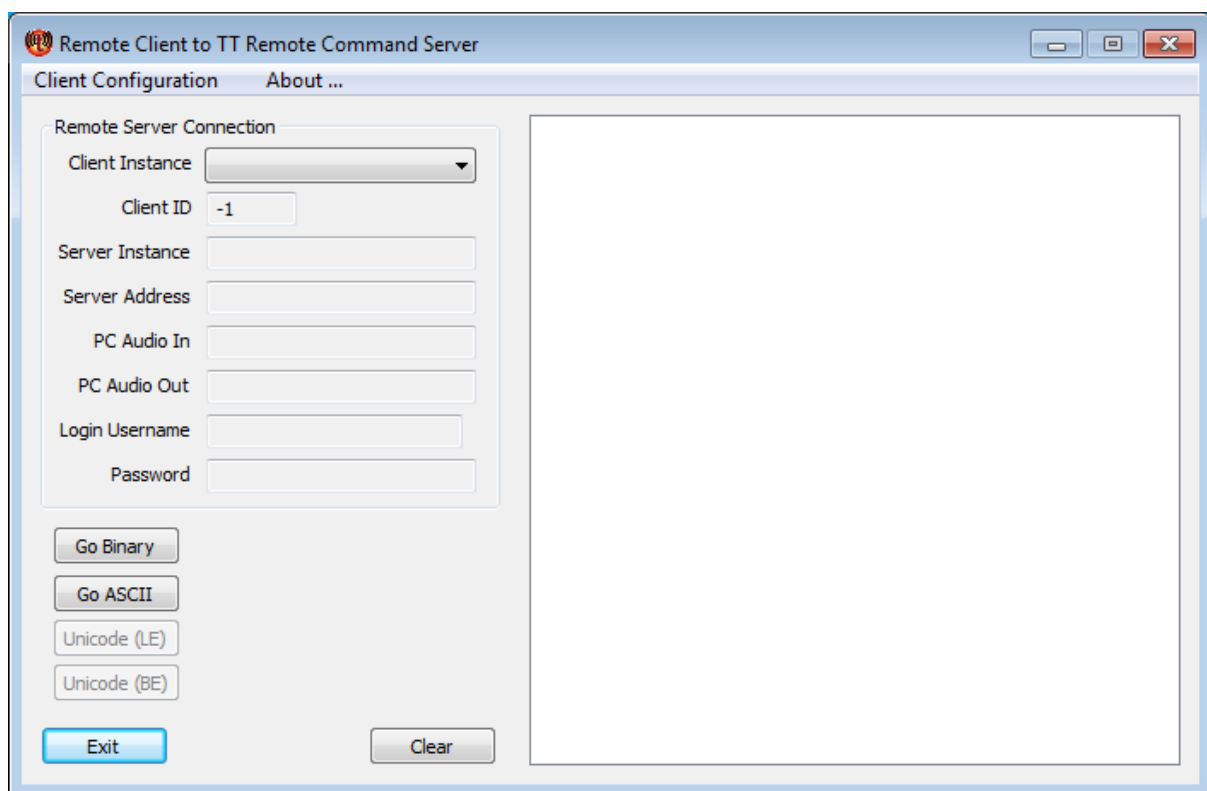
This is just a sample Windows test application to demonstrate how you can control, monitor and use Voice communications with up to 8 Base Station Radios from a remote PC client.

As said, this application is supplied purely to enable experimenters a platform to try various commands, examine the responses and determine an App they want to write will work.

The VS2005 C++ Source project for this client will be published as open source and users will be welcome to use its contents in their own developments. They could even use parts of it as a guideline when developing their own Client Apps - in different coding languages.

Developers will need to have a knowledge of connecting to and communicating with servers using TCP/IP. Also, how to parse text strings our use Network Ordered Packed Binary data packets.

## Initial Launch of TTRemoteClient.exe



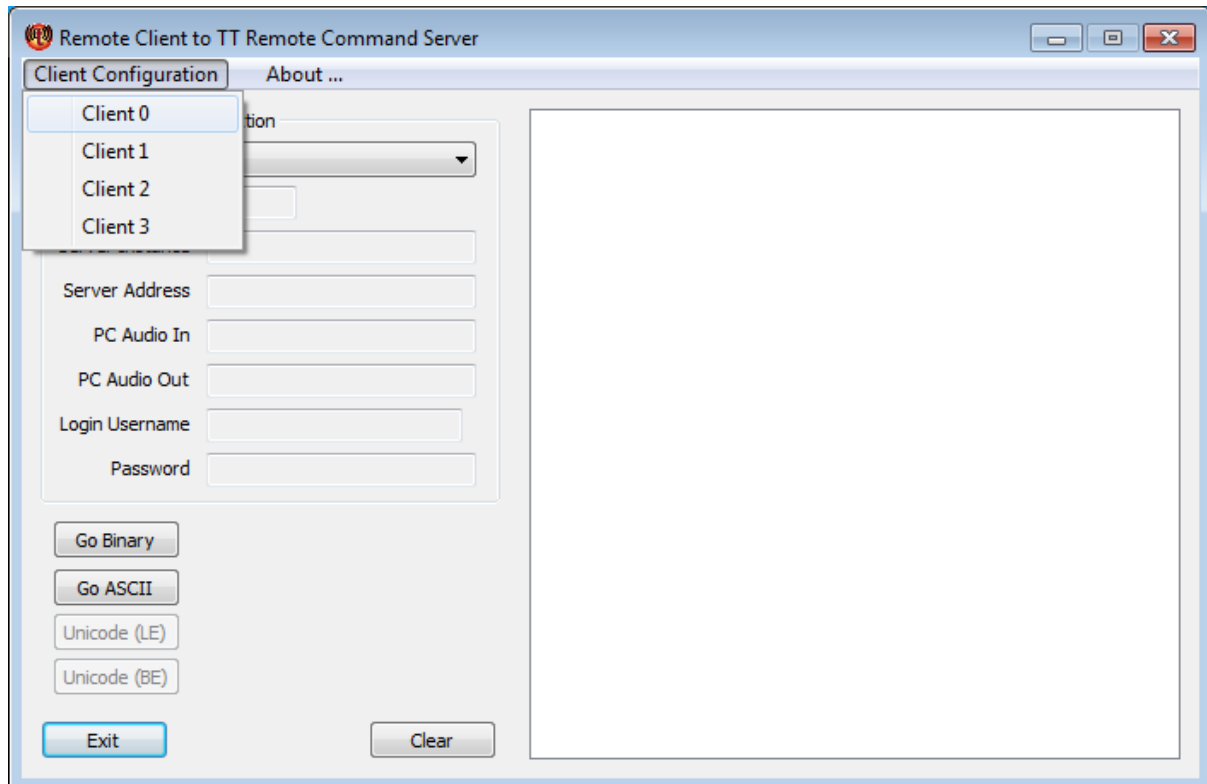
Initial Window will have no Test Clients defined. This version allows you to define up to 4 x clients with different capabilities - 4 is an arbitrary number and should be adequate for testing.

The Menu has 2 x selections, **Client Configuration** and **About**.

Client Configuration will allow you to configure your first Client.

## Configuring a Client

Press the Menu - Client Configuration selection and click the Client 0 entry



This will make the Client Configuration dialog pop up.

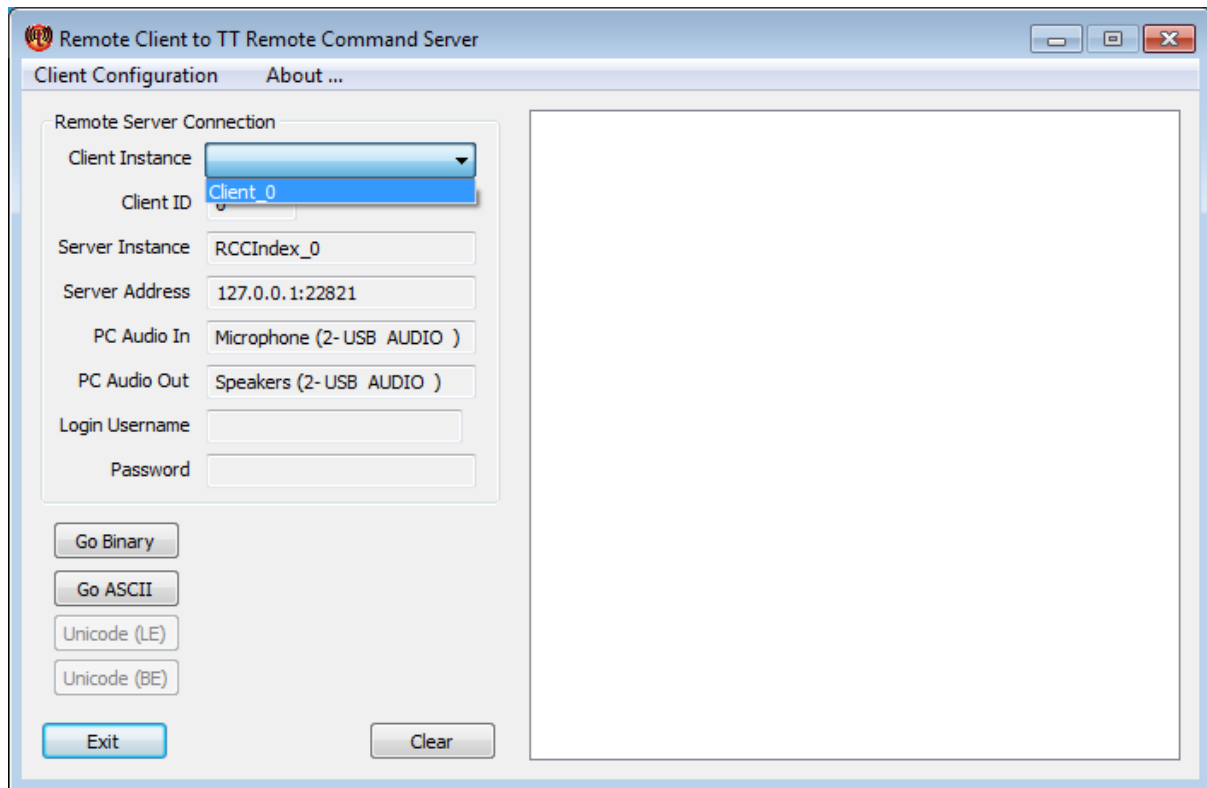
- Select a valid TTServer Radio (RCC) index that was defined with TTHostSetup
- Enter the TTServer IP address and Port
- Select all available check boxes (although currently only a few are implemented)
- If you wish to use Voice select the **PC Audio** radio button and
  - choose the Client's Audio devices

The image shows a 'Client Configuration Dialog' window. At the top, there is a 'Client ID' field with the value '0'. Below this is a section for 'Radio Command Control Server' containing a 'Radio Index' dropdown menu (currently showing 'RCCIndex\_0'), a 'Server Address' field (showing 'RCCIndex\_0'), and a 'Server Port' field (showing '0'). A dropdown menu is open for the 'Radio Index', listing options from 'RCCIndex\_0' to 'RCCIndex\_7'. Below the server settings is a 'Services' section with several checkboxes: 'Radio Status' (checked), 'Display Updates' (checked), 'Call Progress' (checked), 'Voice' (checked), 'ARS' (unchecked), 'Radio Control' (checked), and 'GPS Control' (unchecked). The next section is 'Audio Device Selection', with 'Disabled' and 'PC Audio' (selected) radio buttons. Below that is the 'PC Audio Device' section, which includes 'Input Selection' (set to 'Microphone (2- USB AUDIO )') and 'Output Selection' (set to 'Speakers (2- USB AUDIO )'). At the bottom are 'OK' and 'Cancel' buttons.

When finished press the OK button to save your information.

## Selecting a Client to Test

From the Remote Client Main dialog click the Client Instance dropdown box and click a Client you have defined and wish to test



You have now selected a client to use when testing.

## Notes on the different Client Types

When using the Binary Interface of this test Application, you can only access 1 x RCC Instance (selected) per client, where in fact if you use the Text interface, Telnet or write your own Client Application, you can access any or all of the defined Base Station Radios at the same time.

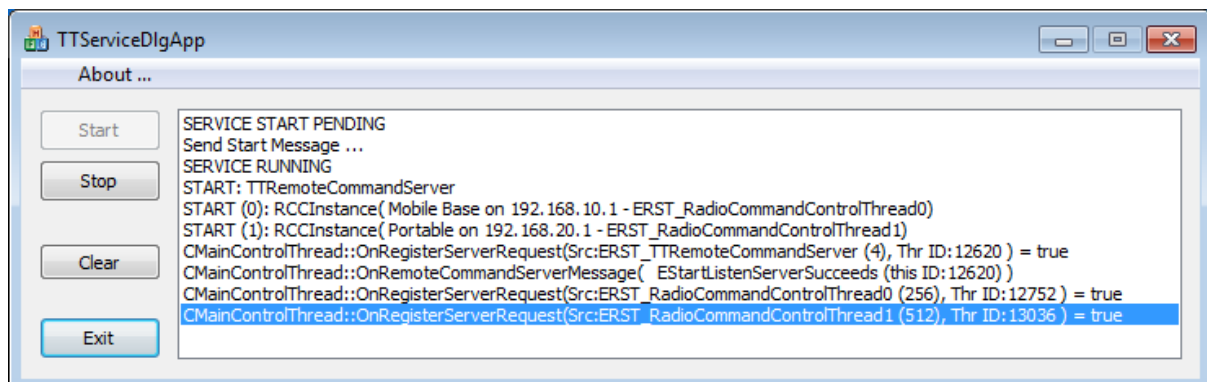
The Client configuration values used here are just for testing purposes and the only real Client requirements are:-

- the IP and port of the TTServer,
- TCP/IP capability and access to the TTServer and
- if Voice is required, then valid local Audio devices.

Even the Protocol (ASCII, Binary, Unicode) is determined by what you send in the first message (TTServer uses the first 2x characters of the first message to determine this). The same with trailing CR, LF or CRLF TTServer checks the last 2 x characters.

For this example we will use the ASCII interface because it allows us to select 1 or more different RCC/Base Station Radios.

☞ Remember to Launch **TTServiceDlgApp.exe** on the server PC or the client will fail to connect.

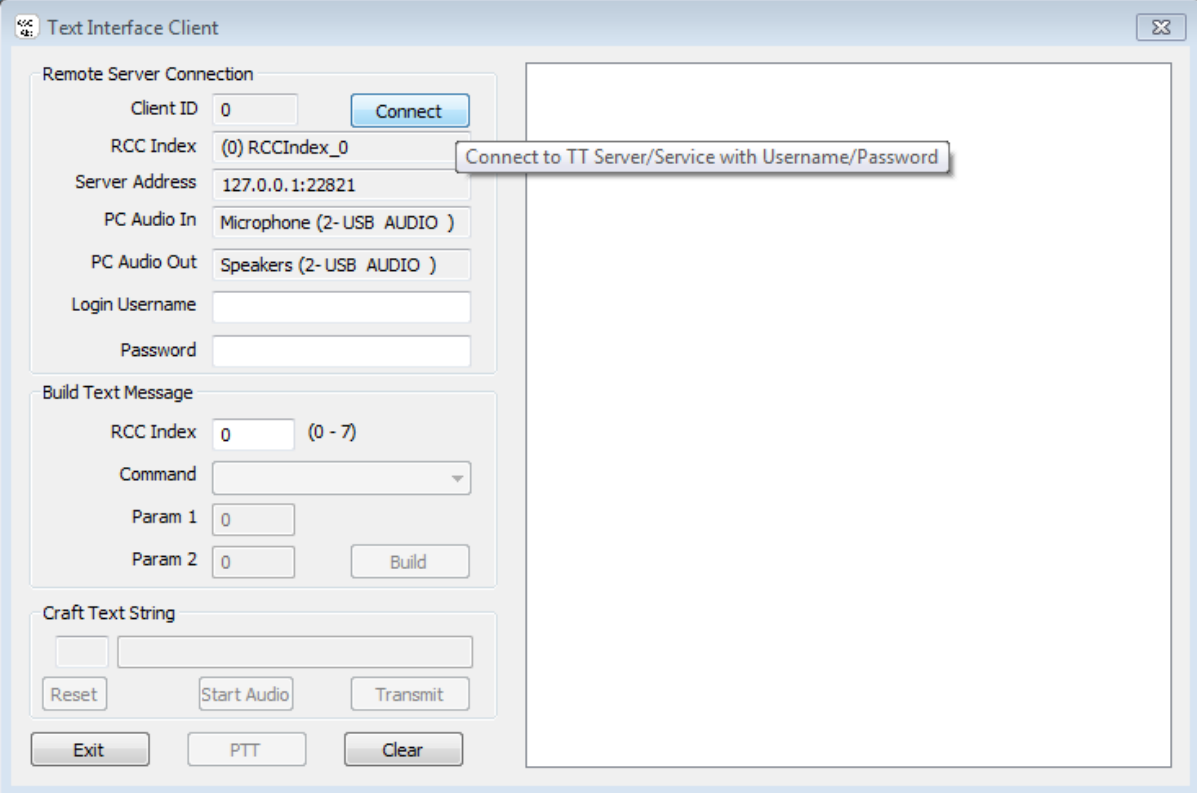


☞ You may run any reasonable (?) number of mixed clients on a single PC. They can be a mixture of Binary and Text - each will be treated as a unique instance by the TTServer and they will only receive information from the specific base stations that they request it from.

## Testing using the ASCII TTServer Interface

Simply press the **Go ASCII** button to launch the ASCII based interface using the selected client as a base reference (remembering this is a work in progress and user feedback may alter its operation).

The Text Interface Client will now be displayed.



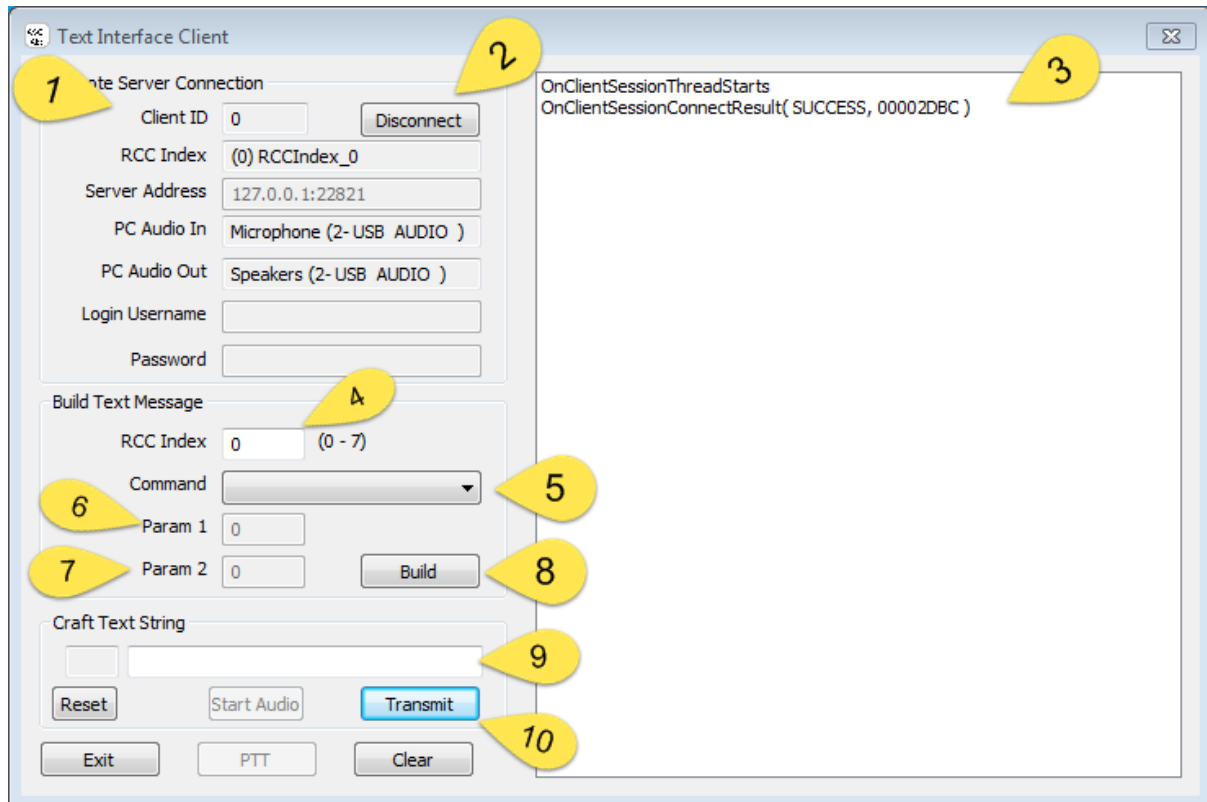
The screenshot shows the 'Text Interface Client' application window. It features three main sections on the left: 'Remote Server Connection', 'Build Text Message', and 'Craft Text String'. The 'Remote Server Connection' section includes fields for Client ID (0), RCC Index ((0) RCCIndex\_0), Server Address (127.0.0.1:22821), PC Audio In (Microphone (2- USB AUDIO)), PC Audio Out (Speakers (2- USB AUDIO)), Login Username, and Password. A 'Connect' button is present. A tooltip 'Connect to TT Server/Service with Username/Password' is visible over the RCC Index field. The 'Build Text Message' section includes fields for RCC Index (0), Command (a dropdown menu), Param 1 (0), and Param 2 (0), with a 'Build' button. The 'Craft Text String' section includes a text input field, 'Reset', 'Start Audio', and 'Transmit' buttons. At the bottom are 'Exit', 'PTT', and 'Clear' buttons. A large empty text area is on the right side of the window.

Ensure the TTServer is running at the IP and port displayed in the **Server address** box.

To connect this client to the TTServer by pressing the **Connect** button.

## Main Points of interest on the Text API Interface (POI)

Ensure that you have connected to the TTserver before proceeding ...



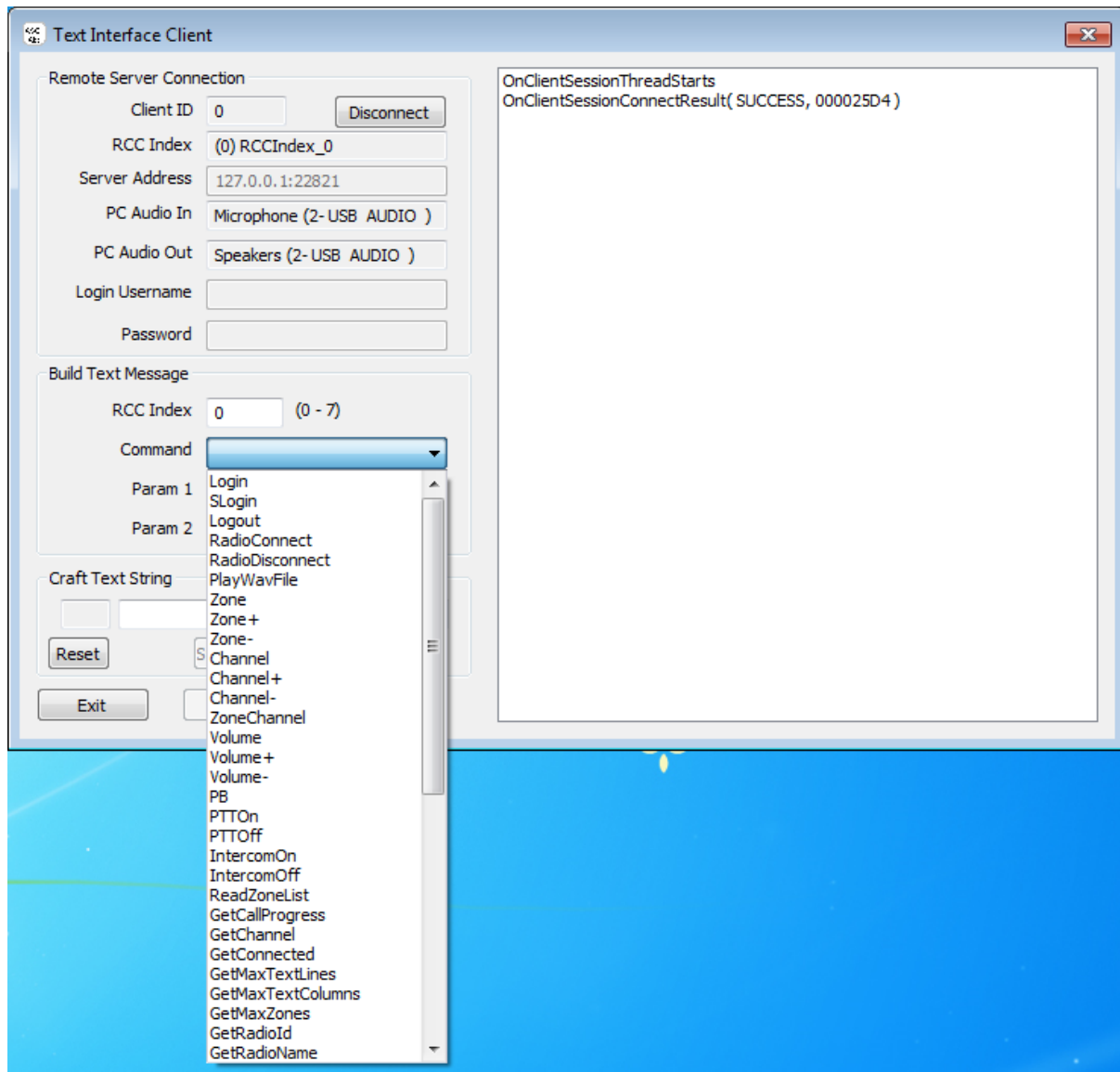
1. The Client Data that was originally selected
2. Connect/Disconnect button for Client to TTServer
3. Output display area - Commands, Responses and Asynchronous messages will appear here
4. The RCC Index for a Base Station Radio defined in TTServer (may be manually changed)
5. Drop down list box for selecting possible commands
6. Entry field for Parameter 1 if required
7. Entry field for Parameter 2 if required
8. Build Button for generating a valid text command string, includes RCC Index, Selected Command, Param1 and Param2 as required
9. The Text string to be sent to the TTServer
10. Transmit Button for sending the built/crafted text message (plus length) to the TTServer

The other Buttons will be addressed later.



## ASCII Radio Command List

Before continuing, it would be worthwhile having a look at **TextMessageFormat.pdf** - as this is where you'll be starting to need it.



As can be seen from this screen shot there are a lot of commands, but the good news is most of them are just simple Radio Control primitive commands.

## Fundamental Commands

For these examples we will assume we will be using the Base Station defined for RCC Index 0.

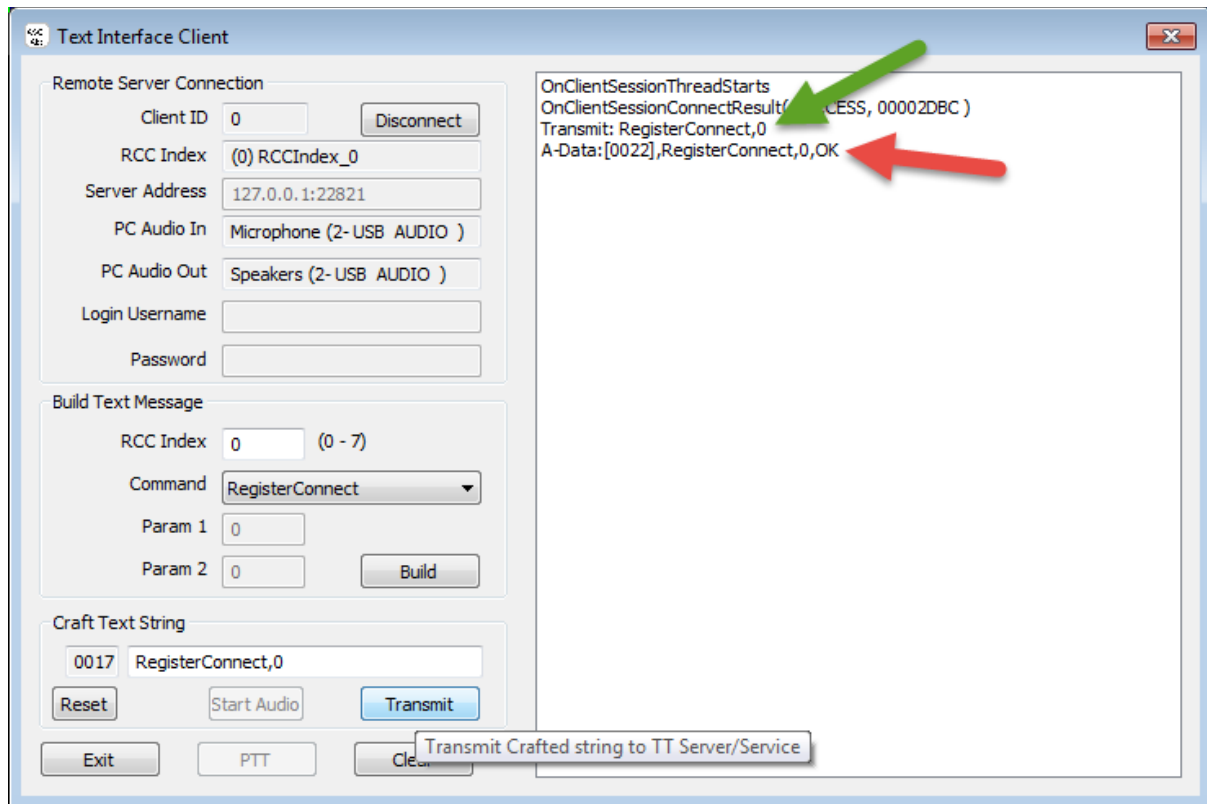
### RegisterConnect - receive connection status changes

One of the most important things required is for the TTServer to be connected to its associated Base Station Radio. Most applications will do this "automatically", they will also detect when the radio gets disconnected due to power loss or some other reason. So if it is critical for your application to know if the Radio is connected (most Apps will), the first command should Register for Connect status changes.

```
RegisterConnect, 0
```

You can type this manually into POI #9 or you can select the command from the Drop Down list (POI #5) press Build Button (POI #8) and examine the result in POI #9.

Press Transmit Button (POI #10) and await the response ..



If your command was successful you will now see your Request, pointed to by the Green arrow and the Response, pointed to by the Red arrow in the Output window.

This means that for all future connection changes to RCC Index 0 TTServer will notify this client.

The following command will stop these notifications:-

```
DeregisterConnect, 0
```

### GetConnected - check TTServer and Radio connection status

Although RegisterConnect notifies you when the Radio's connection status changes - it doesn't tell whether it is currently connected or not.

Two options are immediately available to you

- RadioConnect - which will attempt to connect immediately but it will fail if the radio is already connected - so you'll know, also if its not connected and succeeds then the notification will also reach you
- GetConnected - returns the current state of the connection

Be aware that if more than 1 x LAN client is controlling a Radio - they may all attempt to change its state.

To be continued when I'm not so fed up with doco :(