Submitters Name: Ankan Mookherjee

1) Servers Used
   a) **Ankan Mookherjee**
   b) **Ajay Pantangi**

   Reason:

   a) **Signature:** I wanted a server with a signature which accepted a long and returned a list of strings. List<sting> MethodName(Long)

      Ankan's method: List<sting> GetData(Long).
      Binding: `basicHttpBinding` SOAP Service
      End Point Address: `http://localhost:3355/T9Service.svc/`
      Returns: a list of string with only one string having "-" equivalent to length of long sent.
      Example if no. 1124 is not found it would return "----" and 11 would return "--"
      Contract: IT9Service

      Ajay's method: List<sting> GetWords(Long)
      Binding `basicHttpBinding` SOAP Service
      End Point Address: `http://localhost:44754/T9Service.svc`
      Returns: a list of string with only one string having "-1" if the long sent is not found.
      Contract: IT9Service

   b) **Accessibility:** Ajay is my project partner. He was more accessible. I could easily tell him to fix his error or problems and customize the service to my need which he did. He worked to my satisfaction and gave quick response when I asked him to implement cross domain policy and client accessibility policy. He also changed quickly from wshttpbinding to basichttpbinding when we learnt that former did not work for Silverlight based services.

      I was confident that my server would work, so I used my own server. I was aware of the T9 code that I wrote and reused the controller and model codes in various parts of server for predictive mode and client for non-predictive mode.

      Note: The names of the service T9Service and service contract to be IT9Service are identical for both Ajay and Ankan's service as I asked Ajay to keep it identical. Initially his code had auto-generated code and names. Later I asked it to keep it T9Service for ease of access on server side.

2) List of Files Enclosed:

a) T9Client:

The non-predictive mode was handled at client side only.
The Predictive mode had both Services from Ajay's server and Ankan's server.
Designed and Developed in **Silverlight 4.0**
Contains **Controller Class** to take care of Non-Predicitve functionality.
Contains access to two wcf methods:

i) ServiceReferenceAjay : To reference Ajay's service
ii) ServiceReferenceAnkan: To reference Ankan's service

Only after both services are activated(run in administrator mode) the predictive mode works.

Design:
The T9Client was designed as a Silverlight Application.
It mimics Model View Controller Pattern

i) **Model:** Model.cs
Contains the background data files for non-predictive look up.
ii) **Controller:**MainPage.xaml.cs
Contains the code behind for the MainPage.xaml. It serves as the controller, invokes model and the data from the server via asynchronous calls.
iii) **View:** MainPage.xaml
Contains the design file of the T9Dictionary in the front end.

b) AnkanServerCode: It has the full code and the solution.
Exe file is in **AnkanServerCode\T9Service\T9Service\bin\Debug => T9Service.exe**
There are two folders in the zip file.

i) **T9Service** folder contains the main solution which hosts the wcf service
ii) **T9WcfService** folder contains the wcf service

c) AjayServer: Only the exe files along with the main dynamic link library and other files that he shared. Exe file to be run in administrative mode is **ServerT9.**exe

3) How to Run the Application:

Part A) Host the Web Services

Run Ajay's Server.
To do that Go inside the folder and Run **ServerT9.exe** in administrator mode.

Run Ankan's Server. To do that go inside AnkanServer\T9Service\T9Service\bin\Debug.
Run **T9Service.exe** in administrator mode.

Part B) Run Client Silverlight Application

Go inside T9Client\T9Client\Bin\Debug.
Run **T9ClientTestPage.html**

Radio button in Client

Inside the client, there are two radio buttons. One subscribes to Ajay's service and other to Ankan's service. Whenever either is clicked the service of Ajay or Ankan is set for getting data from Server.

Part C) Exceptions

If you run the client without the server then the client would run predictive mode without problem but would throw message box exception with details when you try to run predictive mode.

Part D) Client Code

Get the Client Code is in **T9Client.sln**
Run in administrator mode
T9Client.Web Project does not contain the changes made to T9Client Project

4) Difficulties and Challenges

a) Cross domain policy and client accessibility policy implementation

b) **Windows 8.1 update:** It made some core changes to Silverlight files. I was unable to debug it as my Silverlight Developer Runtime got corrupted. Multiple tries to set it right did not work. I could not debug it. I could only run the code and see changes to it.

c) Changing from wshttpbinding to basichttpbinding: Initially I did not pay attention to why my service was not being consumed by T9Client. I came to know later that Silverlight does not deal with wshttpbinding. I asked Ajay to change it as well in order to make sure that our services get consumed. He had made the same mistake as well.

d) Use of **System.DateTime.Now.Tick** instead of **System.Diagnostics.StopWatch** in Silverlight client Application: Silverlight does not use System.Diagnostics library which we use for StopWatch class. So in order to manage non-predictive mode, I used Tick function to give me time elapsed between two clicks. 10,000 tick equal to one millisecond. For more than .5 second intervals, we appended the letter or we just changed the letter for the word.