

FrontlineSMS Data Collection without Forms

George Mu' ammar 30/03/2010



Background

FrontlineSMS uses a form for data entry on phones that are java enabled. Unfortunately not all mobile phones are java enabled, and those that are may not be compatible since several java runtime engines exist for mobile phones.

However these phones can send a simple hand-written SMS and FrontlineSMS can receive it, however the data is not interpreted, validated or written in a database by FrontlineSMS.

For this purpose an MS Access database system was developed to host the SMS data and to provide validation on the data received and to send an automatic reply to the sender containing any validation errors found, or to confirm that the data has been accepted. FrontlineSMS interfaces with the database through a DOS script which makes use of some free software utilities.

The database uses SQL instructions and the script interfaces with it using Open Database Connectivity (ODBC) therefore it is easy to port the system onto a different type of database. The backend database consists of one table and can be immediately transferred to another database system (currently Access and MySQL is being used). The front-end database is the system that performs the validation and the queries would require adaptation for porting to another system.

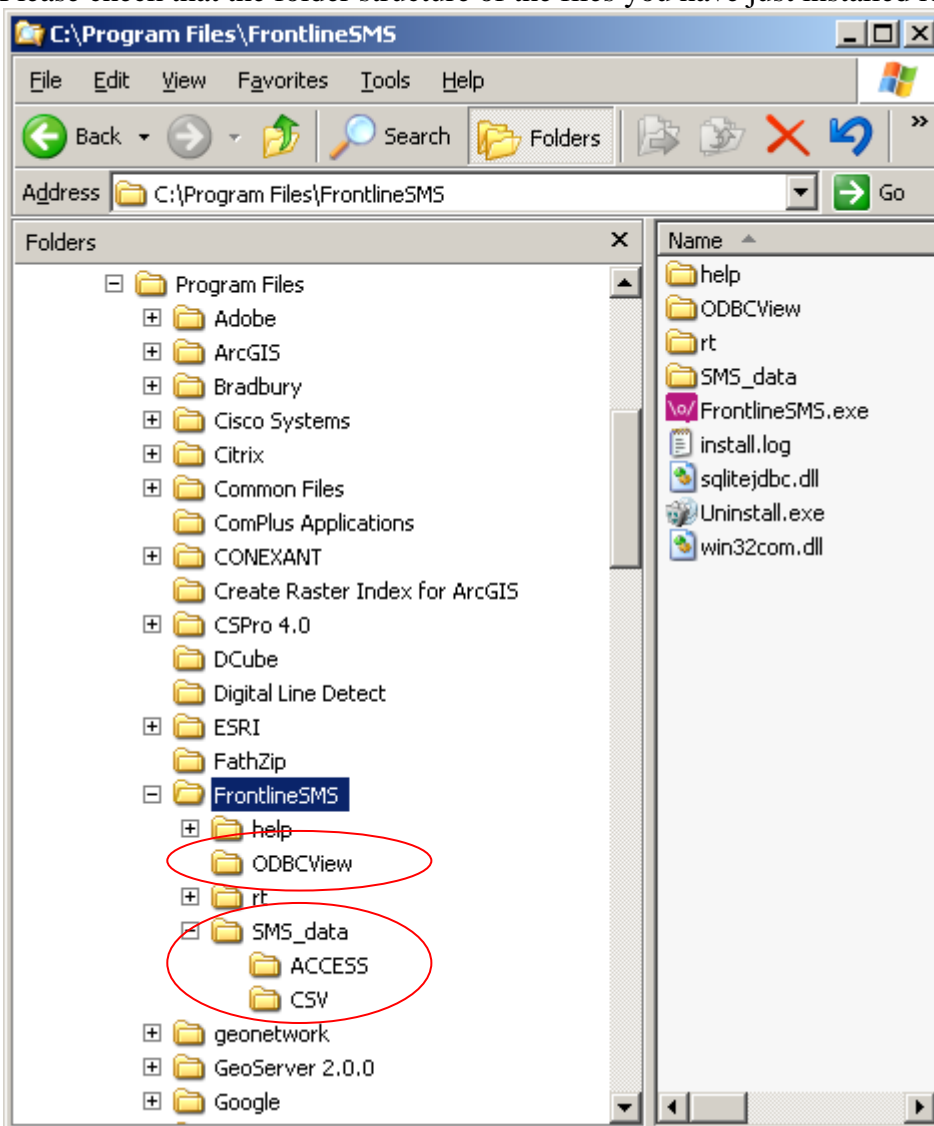
This document describes a setup for an ad-hoc solution. It requires modification to be adapted to another survey. In future WFP will be modifying this system to make a parametric package to be quickly deployed.

| | |
|--|-----------|
| Background | 1 |
| Installation of the SMS system | 2 |
| Setting up the database | 3 |
| Configure FrontlineSMS | 5 |
| Validation | 7 |
| Use of Keywords | 8 |
| Parameters sent by SMS | 9 |
| Table - Validation checks to SMS data | 9 |
| Downloading the Data | 10 |
| Technical Documentation | 12 |
| Software Used | 12 |
| Batch files | 12 |
| Future Enhancements | 14 |
| Modifying the validation queries | 15 |
| Future enhancements: | 16 |

Installation of the SMS system

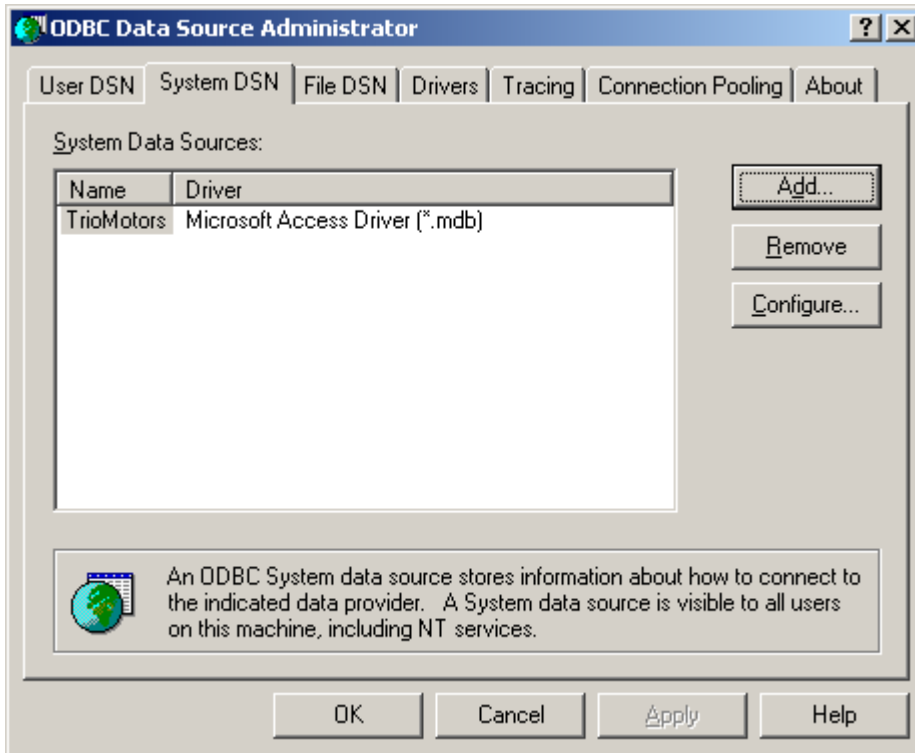
1. Download FrontlineSMS from <http://www.frontlineSMS.com>
2. Run the installation and install into default folder
3. Download ODBCView from <http://www.sliksoftware.co.nz/products/odbcview/>
4. Install to a folder called ODBCView under FrontlineSMS folder. For example C:\Program Files\FrontlineSMS\ODBCView (If you have already installed to another folder, simply copy the existing ODBCView folder and paste it under FrontlineSMS).
5. Copy the folder SMS_Data containing the system described in this text, and paste under the FrontlineSMS installation folder (if you have downloaded the zip file, extract the contents into the FrontlineSMS folder)

Please check that the folder structure of the files you have just installed looks like this:

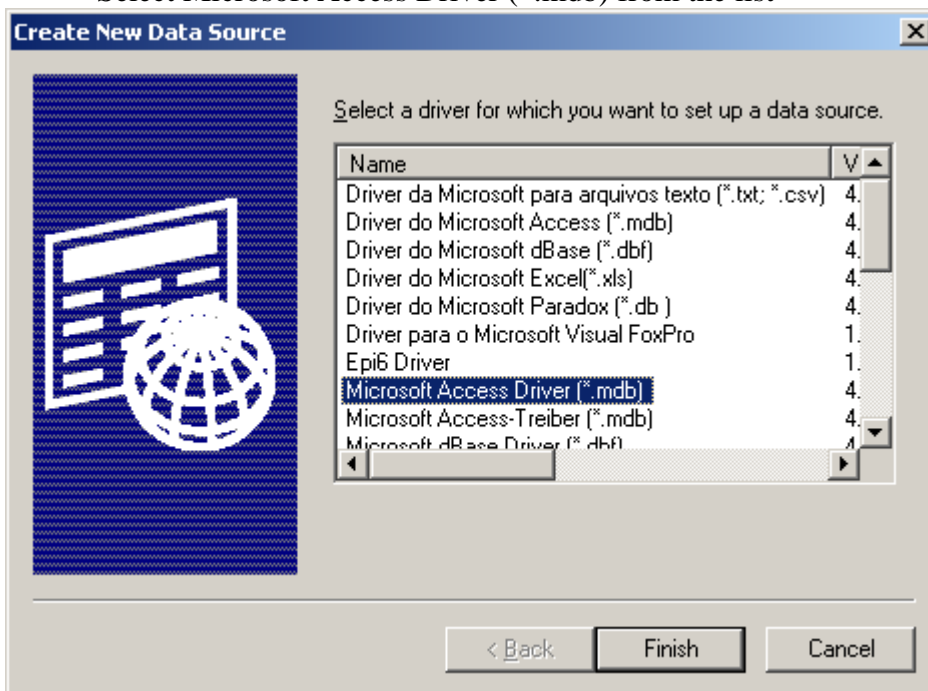


Setting up the database

- Open the ODBC Administrator from START -> Settings -> Control Panel -> Administrative Tools -> Data Sources (ODBC)
- Click on the System DSN tab
- Click on Add

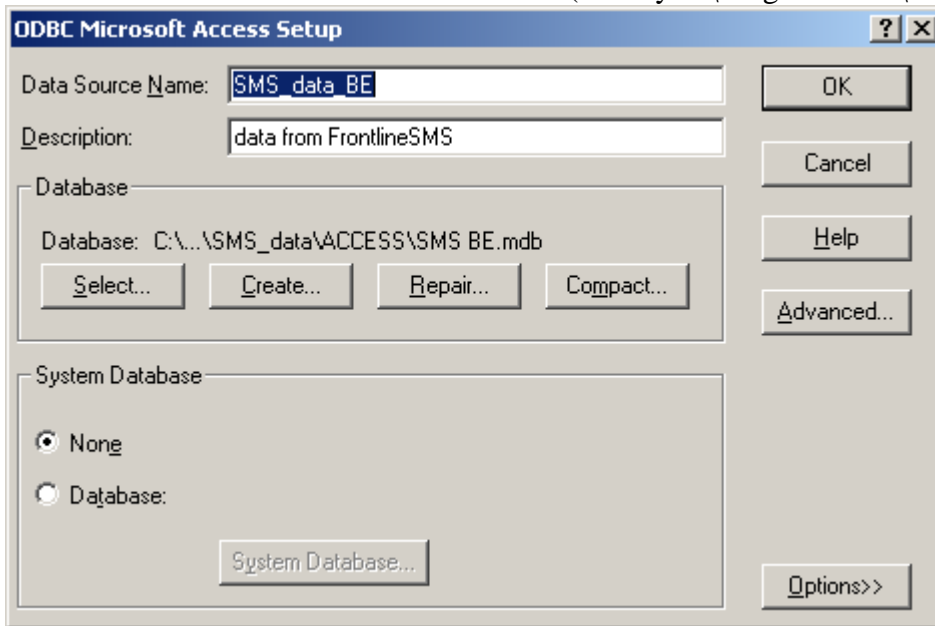


- Select Microsoft Access Driver (*.mdb) from the list

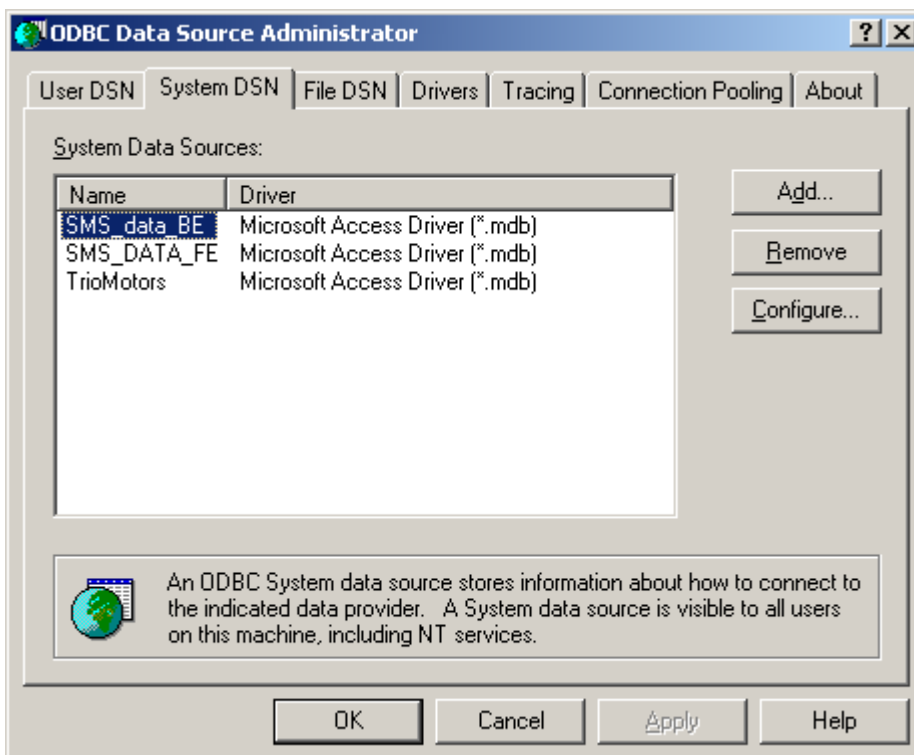


- Specify SMS_Data_BE as the Data Source Name (note: upper or lower case is not important)

- Select to locate the backend database SMS_BE.mdb in the folder SMS_Data\Access under the FrontlineSMS installation folder (usually C:\Program Files\FrontlineSMS)



- Click OK to close this dialog.
- Repeat instructions above to add a second entry called SMS_Data_FE, to point to the SMS_FE.mdb database located in the same folder as above.
- Verify that SMS_Data_FE and SMS_Data_BE are added to the list of System Data Sources and click OK to close it.



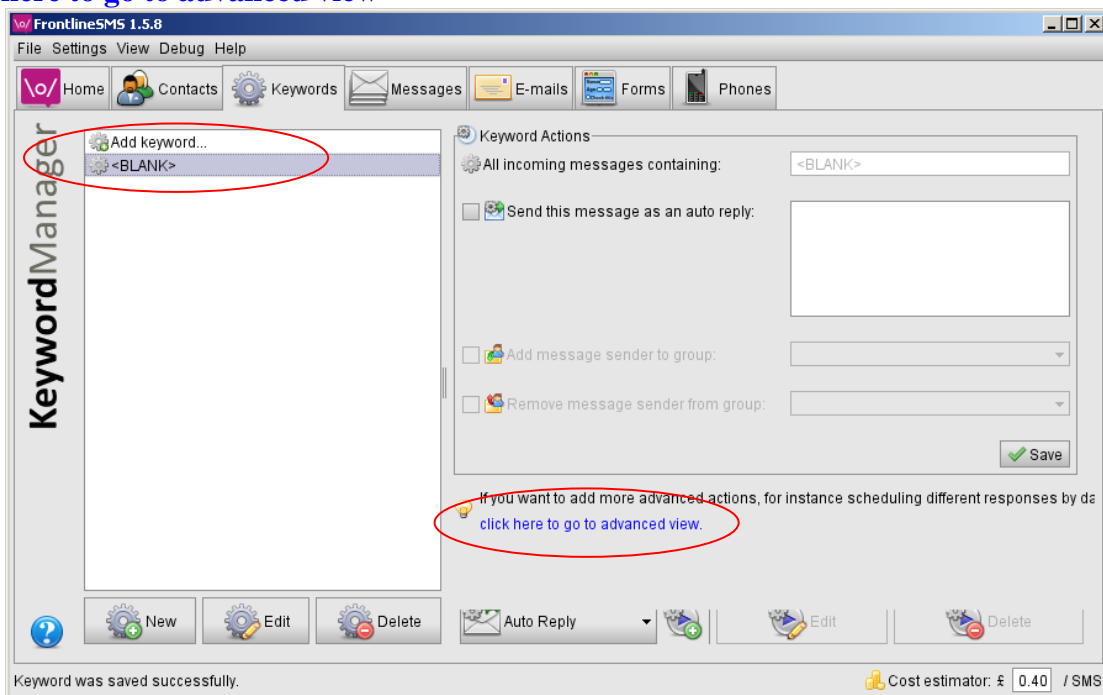
Note: The system is supplied with a MS Access database backend and another MS Access database as a front-end. The backend database can be replaced with any other ODBC-compliant database,

however the SQL requests in the odbc_connection.bat file will have to be adjusted accordingly. The front-end is used for accessing the SMS data and can be replaced with any other system that offers the same functionality.

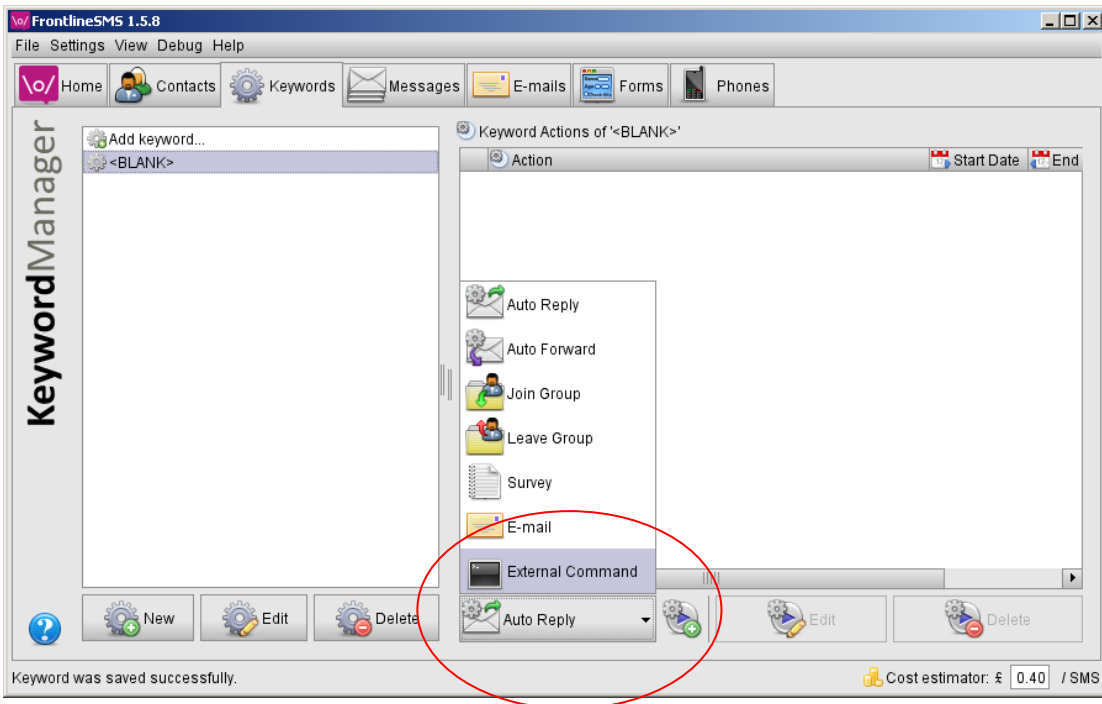
Note: There should be no need to create an ODBC entry to the back-end, since the back-end database table is linked to the front-end in MS Access. This may not be the case for another DBMS.

Configure FrontlineSMS

Under Keywords, Create a new keyword if your data collection uses different keywords with different parameter sets or select <BLANK> if there is only one parameter set and click on [click here to go to advanced view](#)



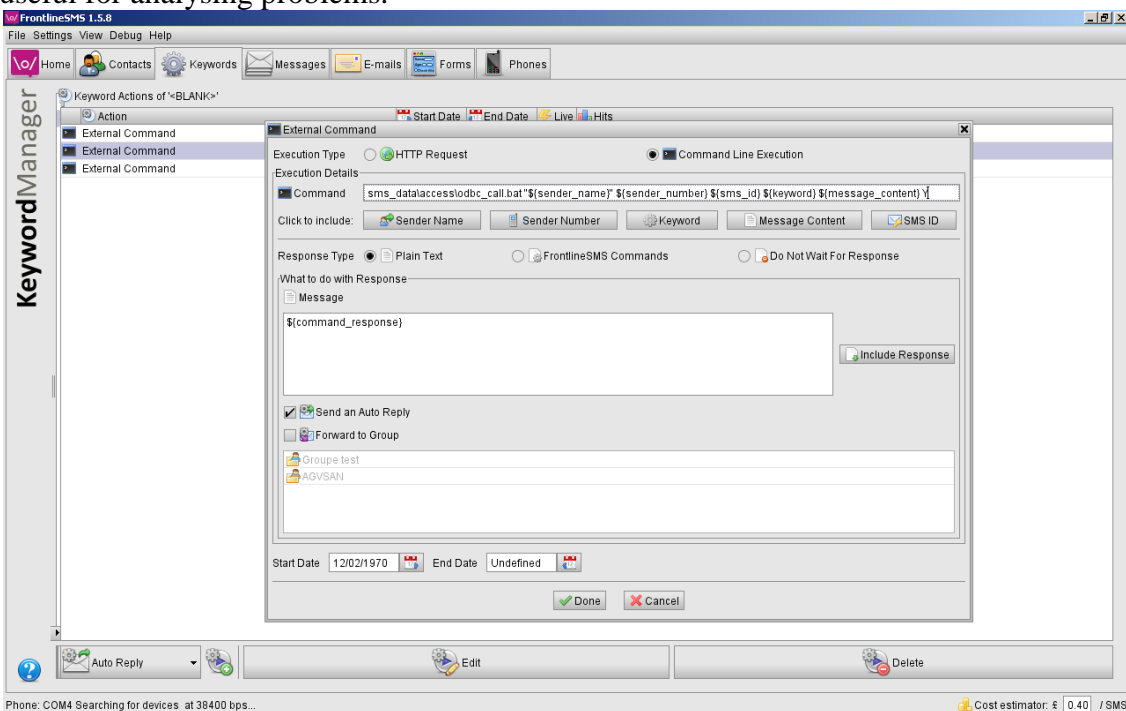
From Auto Reply select External Command, then the arrow button next to it



Make sure to compile the dialog as shown below (or copy and paste from this document): Note the parameters preceded by a \$ symbol in the command line are created using the buttons beneath it. Note the Y at the end is optional and only for debugging purposes, so you should remove it once your system is working.

```
sms_data\access\odbc_call.bat "${sender_name}" ${sender_number} ${sms_id}
${keyword} ${message_content} Y
```

With the optional Y after \${message_content}, a log file will be created called logfile.txt, which is useful for analysing problems.



It is possible to configure a backup system that does not depend on the database, but simply adds all messages received to a CSV file. Create another External Command as above and enter the information shown below: Note no validation is done and therefore there is no response.

Note: again, by adding a blank space and a Y at the end of the Command shown above, a log file will be created for debugging purposes.

FrontlineSMS Contacts

Under Contacts it is advisable to create an entry for each phone sending data. In the validation queries in the database provided there is one validation test for the name of the sender. If the sender name is not present in FrontlineSMS contacts, then there will be an OK reply but will specify that the sender is unknown.

IMPORTANT: Do not put blanks, “, ‘ or + (spaces, double or single quotes, apostrophe or plus signs) in the contact names.

Validation

The following table shows all the possible reply messages that the database will return to FrontlineSMS to be sent as a reply to a message received. For sake of example the messages are referencing in SMS sent for Questionnaire ID 54123 and Household member ID 99. The list is in order of priority since only the first relevant reply will be returned.

The messages that begin with *Erreur* denote an unexpected value for one parameter. The offending value is returned to the user (in place of *<value>*), the name and position of the parameter and the valid range of values are specified.

The messages that begin with OK denote acceptance of the message. This is only relevant for the validation check for contradictory information i.e. if a SMS data message is OK but refers to the same questionnaire ID and same HH member as another SMS data message that was also OK, then the contradictory information validation will fail, and the message will be returned.

All numbers (with exception of questionnaire ID) do not need a leading zero. For example Année Naissance can be specified 4,5,6,..10 representing 2004...2010 (for children age 0 to 5).

Decimal point can be represented either with . (dot) or with , (comma). The database front-end will show the parameters with a dot or a comma depending on the regional settings of the PC on which the front-end is installed. The backend will archive the SMS with the original dot or comma as sent by the enumerator.

The database front-end will validate the data SMS received, return the first appropriate reply to FrontlineSMS to send as a reply SMS, and will archive the SMS data together with the reply message in the backend.

Please refer to the technical annex regarding how the validation system works. Some validation messages return a skip code to the user, which is a number that the user can add at the end of their message to avoid that validation check.

Use of Keywords

Note that in the current version keywords are not being used. Keywords would enable the reception of different sets of parameters by SMS. This was not requested at this time and was not implemented. It is expected to be implemented in future versions due to the simplicity of the upgrade required.

To use Keywords, create a query instead of `Keywords_All` and add the condition
... `WHERE keyword="keyword"`

Then use this query in the place of `Keywords_All` for your validation.

Parameters sent by SMS

The parameters to be sent by SMS, as are:

- 0.0a N° du Questionnaire
- 1.1 Code du membre dans le ménage
- 1.2 Sexe
- 1.6 Date de naissance de l'enfant (mois et année)
- 1.9 Poids en kg
- 1.10 Taille en cm
- 1.11 Tour de bras en cm
- 1.12 Œdèmes bilatéraux

Optionally: the user can add a skip code: a number to skip all validation checks with priority <= to the supplied skip code. See the technical annex on validation
Also the user can add a message up to 16 words at the end of the data in the SMS

Table - Validation checks to SMS data

The validation checks automatically made on the SMS data sent are the following, in order of priority.

| |
|--|
| Erreur Q=54123 M=1 Nombre de parametres est insuffisante |
| Erreur Q=54350 M=50 No. du Questionnaire (P 1) <value> pas entre 1001 et 54250 |
| Erreur Q=54123 M=51 Code Membre (P 2) <value> pas entre 1 et 50 [P10 = 4] |
| Erreur Q=54123 M=50 Sexe (P 3) <value> pas entre 1 et 2 |
| Erreur Q=54123 M=50 Age en mois (P 4 & 5) <value> pas entre 6 et 59 |
| Erreur Q=54123 M=50 OEdemes Bilateraux (P 9) <value> pas entre 0 et 1 |
| Avertissement Q=54123 M=1 Tour de bras (P 8) <value> pas entre 6 et 20 [P10 = 11] |
| Avertissement Q=54123 M=50 Rapport Poids Age (P 4,5 & 6) <value> pas entre 6.7 et 14.1 de l OMS [P10 = 20] |
| Avertissement Q=54123 M=50 Rapport Taille Age (P 4,5 & 7) <value> pas entre 69.3 et 85.7 de l OMS [P10 = 21] |
| Avertissement Q=54123 M=50 Rapport Poids Taille (P 6 & 7) <value> pas entre 12.1 et 20.9 de l OMS [P10 = 22] |
| OK* Q=54123 M=50 mais Numero de l equipe dans le QuestionnaireID <value> ne correspond pas a le propriétaire de le telephone <nom> |
| OK* Q=54123 M=50 mais Expediteur inconnu |
| OK* Q=54123 M=1 mais donnees contradictoires avec sms precedent des 12/03/2010 0:20:08.45 (<nom>) |
| OK Q=54123 M=50 |

Downloading the Data

The data is physically stored in the back-end database called SMS_Validated. This contains all the SMS data and the response that was sent back to the enumerator.

However the data is best viewed by accessing the front-end database query called KEYWORDS_ALL_VALIDATED.

This query adds the field names to each field and corrects the decimal point or comma issue.

Accessing the SMS Data

The SMS data is accessible from the database front-end

C:\Program Files\FrontlineSMS\SMS_Data\ACCESS\SMS_Data_FE.mdb (a link to this file is placed on the desktop)

When opened, a Form will open automatically for exporting the data.

The user can select the name of a data analyst. The date of the last export will appear underneath, and a preview of the data received after that date.

The screenshot shows the Microsoft Access interface with the 'FormExport : Form' open. The form contains several fields and a data table. Callouts provide the following information:

- Select User name here:** Points to the 'DataAnalystName' dropdown menu, which currently shows 'B'. A 'Manage User' button is located to the right.
- Dates of previous exports are listed:** Points to the 'Last Exported SMS' dropdown menu, which shows a list of dates including '07/03/2010 19:21:41', '07/03/2010 19:21:41', and '01/01/2001 00:00:00'.
- A preview of the data to export is listed:** Points to the 'Data to export' table.
- The filename of the file to be produced:** Points to the 'Filename' text box, which contains 'SMS_B_201003182226.xls'.
- The last SMS to be exported. This is remembered by the system for the uses next export:** Points to the 'last SMS to be exported' dropdown menu, which shows '14/03/2010 20:02:25'.
- Click to export the data to MS Excel:** Points to the 'Export' button.
- User management. See below:** Points to the 'Manage User' button.

| ID | Sender_name | Sender_number | SMS_date | SMS_time |
|-----|-------------|-----------------|------------|-------------|
| 172 | 35 | %2B221774462606 | 14/03/2010 | 14:04:01.35 |
| 175 | 34 | %2B221775371684 | 14/03/2010 | 14:14:38.53 |
| 176 | 34 | %2B221775371684 | 14/03/2010 | 14:16:34.67 |
| 177 | 35 | %2B221774462606 | 14/03/2010 | 14:17:43.39 |
| 179 | 34 | %2B221775371684 | 14/03/2010 | 14:20:30.12 |
| 180 | 35 | %2B221774462606 | 14/03/2010 | 14:24:49.75 |
| 181 | 33 | %2B221775721295 | 14/03/2010 | 14:27:56.71 |
| 182 | 33 | %2B221775721295 | 14/03/2010 | 14:30:38.79 |
| 183 | 32 | %2B221776138443 | 14/03/2010 | 14:38:32.64 |
| 184 | 33 | %2B221775721295 | 14/03/2010 | 14:43:49.29 |
| 185 | 32 | %2B221776138443 | 14/03/2010 | 14:46:56.39 |

Technical Documentation

Software Used

FrontlineSMS is a product made by Kiwanja.net. Please refer to the website www.FrontlineSMS.com for more information. Keep in mind that FrontlineSMS is made for data collection using java enabled phones, and using the Forms functionality. In this document we are describing a system for non-java enabled phones, using the *Keywords* functionality of FrontlineSMS.

MS Access is the database application in MS Office. MS Access is used as a front-end and as a back-end database through ODBC queries. It is easily replaceable by other database systems that have an ODBC driver. A custom application has been written and is freely available with this text.

ODBCView is a free utility produced by SLIK Software (www.slik.co.nz). It is used as an ODBC interface.

Change is a free DOS utility included in the download (originally downloaded from <http://www.golden-triangle.com/CHANGE.ZIP>)

QGREP is a free DOS utility included in the download, produced by Microsoft and available with the Windows 2003 Server Resource Kit.

The batch file interface between FrontlineSMS and MS Access using the utilities has been custom made and is freely available with this text.

Batch file

When FrontlineSMS receives an SMS it will call a batch file passing appropriate parameters. The batch file for passing data to the MS Access database is called ODBC_CALL.BAT and resides in the ACCESS folder inside the zip file. The functions of this batch file are summarised as follows:

1. Parse the parameters sent by FrontlineSMS, adding NULL to bring the count of parameters to 10.
2. Empty a temporary table in the front-end database using an SQL query passed to ODBCView
3. Create an SQL query to insert the SMS parameters into the database
4. Call ODBCView using this query to add the SMS parameters to the front-end database temporary table
5. Call ODBCView to run the validation queries on the SMS parameters in the temporary table
6. Use the validation response and the same SMS parameters to create another query to upload the data with the response to the back-end database
7. Return the response as a result of the batch file to the standard output (which will be used by FrontlineSMS as a reply SMS)

There are 3 configuration parameters at the beginning of the batch file to be set for proper operation. Edit the batch file using a text editor and make sure they are correctly configured. They are:

1. The directory on disk where the script is located
2. The ODBC DSN of the front end and the backend

3. An existing folder where to backup the SQL commands used to enter the data. Set to NO if this is not required
4. The number of mandatory parameters in the SMS. This is not yet implemented, and is currently set to 9

The following support files are created by the batch file:

| | |
|-----------------------|---|
| LOGFILE.TXT | if the Y option is used at the end of the command line in FrontlineSMS then all SQL commands and the output of all operations will be logged to this file |
| Delete1.sql | SQL statement to empty temporary table in front-end database |
| Temp.txt | Temporary file to capture unnecessary output |
| InsertInto1.SQL | SQL statement to upload the SMS parameters to the front end for validation |
| InsertIntoOutput1.txt | The output of the above SQL query |
| Validate.SQL | SQL statement to run the validation queries and obtain response |
| ValidateResult.txt | The result of the above operation |
| ValidateOut.txt | The processed result for use in following SQL query and as a reply SMS |
| InsertInto2.sql | The SQL statement for uploading the final SMS data in the back-end database |
| InsertIntoOutput2.txt | The result of the above operation |

The ODBC_CALL.bat batch file is fully documented, and may be modified as required. Note that any output or errors produced by this file will be returned as a reply SMS by FrontlineSMS. Therefore it is important to capture all output created by any command, except the final command of typing the contents ValidOut.txt as a reply SMS. Please make sure to note the usage of **>temp.txt 2>&1** after the end of DOS commands to eliminate any outputs or errors.

The command line usage of the batch file is as follows:

ODBC_CALL <sender_name> <sender_number> <SMS_ID> <keyword> <parameters> <logfile flag>

Where

| | |
|-----------------|---|
| <sender_name> | is the name of the sender to be entered in the database |
| <sender_number> | is the number of the SMS sender |
| <SMS_ID> | appears not to be used by FrontlineSMS |
| <keyword> | is the keyword that triggered the batch file call, currently not used |
| <parameters> | is the list of parameters in the SMS separated by + (plus) |
| <logfile flag> | is the letter Y if a logfile is requested |

The batch file is called by FrontlineSMS so the current folder is the installation folder of frontlineSMS (C:\program files\frontlineSMS for example). An example of a call to the batch file is as follows:

```
SMS_data\ACCESS\odbc_call.bat George 123456789 1 keyword
54123+11+2+2+9+12%2C3+111%2C8+22%.9+0 Y
```

Note that FrontlineSMS sends a %2C instead of a , (comma)

The backup system is also based on a batch file called APPEND.BAT. The functioning is much simpler since it simply converts the + delimiter used by FrontlineSMS into a CSV delimiter and add

the text to the end of the file DATA_EN.csv for English systems (using a , (comma) delimiter) and to the file DATA_FR.csv for French or other regional systems that use a ; (semi-colon) as delimiter. The simplicity of this system is important to ensure that it will not fail if an unexpected situation arises that may cause the database system to fail.

Future Enhancements

An error control mechanism should be built to capture ODBC results and to control possible errors in order to ensure that an acceptable error is always sent to the SMS user, and that the data is always uploaded into the database even if validation queries do not work. Currently if an error occurs with the ODBC or with the database queries then the ODBC error message will be sent by SMS as a reply (which will be incomprehensible to the user), and no data will be uploaded into the database.

Known bugs: Contact name in FLSMS cannot contain a space. Quotes should be used around the sender_name parameter in the batch file and FLSMS command line to allow this.

The Database System

The backend database system consists only of a table named SMS_Validated and contains the SMS data and the reply SMS that was sent to each SMS.

The front-end database system contains the table where the SMS parameters are initially uploaded, the validation queries, a link to the back-end SMS_Validated table and a query for producing the final results to the user. Forms and reports will soon be created to further enhance the end-user experience.

Following is a list of tables and queries used in the front-end database:

| Table Name | Description |
|----------------------|---|
| SMS_Data | Temporary table used to contain the SMS parameters for validation |
| SMS_Validated (link) | A link to the table in the back-end |
| Validity_Params-test | UNUSED: A table for testing a new validation system being envisaged. |
| Query Name | Description |
| Keywords_All | Adds field names to the parameters received by SMS |
| Keywords_Validated | This query should be used by the data analyst since it will contain all the parameters, field names and validation responses in the database. |
| Validation_FullCheck | Appends all the Validate_NN_desc validation queries regarding the data in the SMS_Data table into a single query listing all responses in order of priority |
| Validate_NN_Desc | Contains each single validation query to be run on the data. These have a standard format and will be put together by Validation_FullCheck |

Creating custom validation queries

The validation queries require SQL query knowledge. The queries check for one error each and respond with an appropriate error message (or ok message). Each query has a priority ranking. The queries are put together by Validation_All which is run by Validation_FullCheck which removes validation checks as defined by the skip parameter supplied by the user (to skip validation).

Each validation query must contain the following fields:

- Check: must have the value TRUE if the validation is passed or FALSE if it fails
- Response: the text message to be sent to the sender if validation fails
- Priority: a number that defines an order of importance. Lower is more important
- Sender_Number: all following fields are taken from the SMS_Data record as is
 - Date
 - Time
 - ID

The validation queries represent the delicate point of the system. The user must bear in mind that the SMS parameters received are upload as text fields, therefore some issues are to be considered when checking for validity. These are listed below:

1. The field value should not be converted directly to numerical because this can generate errors if an attempt to convert an illegal value is performed
2. The user may include a comma or a dot for a decimal value
3. The IsNumerical function should be used to test for numerical validity
4. > and < tests behave differently for strings as they do for numbers. For example “16”<”2”. The solution is to zero pad numbers on the left to bring them to the desired length.

Please see Validate_10_Muac as an example of how to check if a parameter is a number and has a decimal point and is within a specified range. See Validate_20_WHO_WAZ for an example of how to compare data sent with data in another table to obtain validation information

It is up to the develop to construct the response text. Where relevant the response text may include the priority of the validation query. If the user re-sends their SMS adding an extra parameter, this will be used to skip all validation checks with priority \leq to the value of the extra parameter (skip code).

Future enhancements:

- The validation system is delicate in as much that if a user makes a mistake in the sql query, an ODBC error will be returned. A system should be made to capture SQL errors.
- Validation is clumsy and requires advanced SQL knowledge. A system that allows a user to enter validation information into a table is being envisaged and tested but not yet functional see table Validity_Params-test and the VBA function Validation()