DISTINCTIVE FEATURES OF MOBILE MESSAGES PROCESSING

Ken Braithwaite, Mark Lishman, Vladimir Lovitskii, David Traynor

Abstract: World's mobile market pushes past 2 billion lines in 2005. Success in these competitive markets requires operational excellence with product and service innovation to improve the mobile performance. Mobile users very often prefer to send a mobile instant message or text messages rather than talking on a mobile. Well developed "written speech analysis" does not work not only with "verbal speech" but also with "mobile text messages". The main purpose of our paper is, firstly, to highlight the problems of mobile text messages processing and, secondly, to show the possible ways of solving these problems.

Keywords: mobile text messages, text message analysis, natural language processing

ACM Classification Keywords: I.2 Artificial intelligence: I.2.7 Natural Language Processing – Text Analysis

Introduction

1. The reasons why is very difficult to use the classical linguistic approach for verbal speech analysis have been considered in [1]. In this paper the problems of Mobile Short Message (MSM) analysis will be discussed. MSM represents plain text message of 160 characters or less and provided by mobile SMS (short message service). The year 2005 saw an explosion in the volume of MSM being sent to mobile phones. Mobile's users choose to send MSM rather than talking on a mobile call because [2]:
   - ♦ They don't have time to chat phone (74%).
   - ♦ To not disturb other patrons on public transportation or at a sporting event or restaurant (53%).
   - ♦ To get work done and send quick notes when on the road travelling for business (32%).
   - ♦ Less disturbing than phone calls (72.5%).
   - ♦ One can reach the other party around the clock (30.4%).

   However, mobile operators need to understand that subscribers give greater priority to the convenience of using the service over the technology and capabilities it offers. Therefore, more effort must be placed on creating user-friendly client interfaces that integrate effectively with the handset features.
2. A wide variety of information services can be provided by SMS, including weather reports, traffic information, inventory management, itinerary confirmation, sales order processing, asset tracking, automatic vehicle location, entertainment information (e.g., cinema, theatre, concerts), financial information (e.g., stock quotes, exchange rates, banking, brokerage services), and directory assistance. SMS can support both *push* (i.e. mobile-terminated (MT)) SM and pull (i.e. mobile-originated (MO)) SM to allow not only delivery under specific conditions but also delivery on demand, as a response to a request.
3. The important distinctive feature of MSM is that the majority of them are bilingual (i.e., using both English words and mobile slang from Tegic's T9 dictionary [3]).
4. We will consider MSM in indissoluble link with Inbound Number (INo) represented by a short code (it is typically a 5 digit number which is accessible by subscribers of any mobile operator) or long code (a usual mobile number– works across all operators).
5. Information services as described above are provided by "Content Providers" who must rent an INo. This can be dedicated to provide a single service or shared to provide multiple services. In they case of multiple services, they are distinguished by the use of a key word that user must provide as the first word of the MSM.
6. The standard 12-key keypad found on many mobile phones today (see Figure 1). On this Figure "Imitator of Mobile" is represented. Alphabetic letters are mapped to keys '2' through '9'. However, this arrangement poses problems for text entry. As three or four letters share the same key, some form of disambiguation is required to determine which letter is intended by the user. There are currently two main methods that are

usually used on mobile phones for text entry. They are the multi-tap method and the predictive text entry method. In the multi-tap method, a user taps the key that contains the letter repeatedly until the desired letter appears. The number of taps required depends on the position of the letter on the key. In predictive text input method (e.g., Tegic's T9 [3]), the user presses the key that corresponds to each letter of a word once. The system uses a dictionary of words to determine which of the possible words the key sequence matches. When MSM is received on a particular INo, then for a dedicated INo the MSM is forwarded to the client renting it. If the INo is shared, the MSM needs to be examined to identify the client and the individual service.

7. First we will describe the types of MSM and the problems encountered examining the MSM. The MSM might be represented by:

- Letter or digit. For example, a number of promotions are quizzes/competitions and sometimes are also interactive, i.e., multiple messages/responses. If the original message to the customer is a question, such as *"How many legs has my dog got?"* then the customer could reply *1*, *2*, *3*, or *4*. Some promotions are multi-choice answers e.g., *'a'*, *'b'*, or *'c'*.
- Single word or number (e.g. credit card number).
- Sequence of words or numbers.
- Combination of words and numbers in MSM.

The main purpose of this paper is to investigate the bad pairs INo ↔ MSM and find ways to restore them.

Figure 1. Standard 12-keys keypad

Let's call pair INo ↔ MSM *bad* if:

- INo does not exist;
- Type of MSM was not recognised or keyword of MSM was not recognised. Very often the first whitespace-delimited word represents keyword (KW) and allows the identification of the client;
- The pair INo ↔ MSM does not exist because $(\neg INo \,\&\, MSM) \vee (INo \,\&\, \neg MSM)$,

  where ¬INo and ¬MSM stand for *wrong* INo and *wrong* MSM respectively. Let's call INo and MSM *wrong* if they separately exist but link between INo and KW of MSM does not.The reason of wrong MSM is understandable. For example, a user can tap the 2-key once to get 'a', twice to get 'b' and thrice to get 'c'. If he taped wrongly then instead of desired word *bell* he typed *cell*, or using 6-key instead of *come* was *cone*.
- A special type of MSM (so called ***stop MSM)*** requires synonyms for recognition e.g., *cancel*, *remove*, etc.
- Finally, we would like to underline the most difficult and dangerous problem when INo ↔ MSM exists but

  $((INo^T \neq INo^D) \,\&\, (KW^T = KW^D)) \vee ((INo^T = INo^D) \,\&\, (KW^T \neq KW^D)) \vee ((INo^T \neq INo^D) \,\&\, (KW^T \neq KW^D))$,

  where letters *D* and *T* mean what user *desired* to type and what was actually *typed*.

This problem takes place because of ambiguity of both INo and KW i.e., one INo might link to several KW and many different INo might use the same KW, and vice versa.

Let's investigate these problems and discuss the results of KW, INo and bad MSM analysis. Our investigation was grounded in real data analysis. As a result of this discussion an algorithm to deduce the correct KW from a bad MSM will be described. Also, the result of using of this algorithm will be shown.

Keywords Analysis

The result of KW analysis and KW ambiguity is shown on Figure 2, namely:

- Total (valid + invalid) KW distribution among letters and mobile's keys (2-9). A KW is *invalid* if it currently is not used on the INo but at the same time the same KW might be valid for another INo. For example, KW *red* is valid for INo 81025 and 80039, and invalid for 89095;
- Displaying the list of KW for selected letter or Inbound No by clicking the corresponding letter or digit;
- For any KW (by clicking when the list of KW is displayed, or just simply typing in KW) the corresponding list of INo is displayed.;
- List of the next (= expected) symbols is displayed for the entered symbol (letter or digit);
- List of ambiguity for both valid and invalid KW is displayed.
- INo ambiguity is shown on Figure 3.

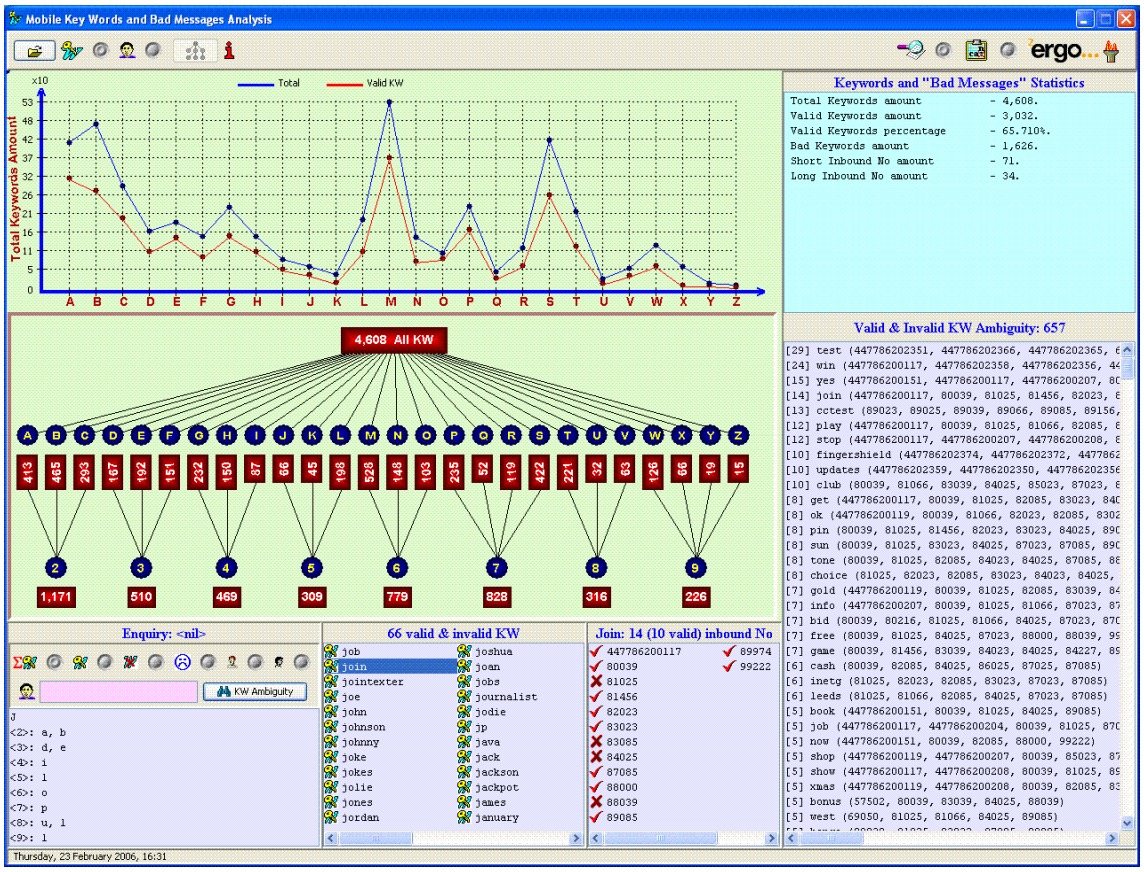

Figure 2. Keywords analysis and KW ambiguity

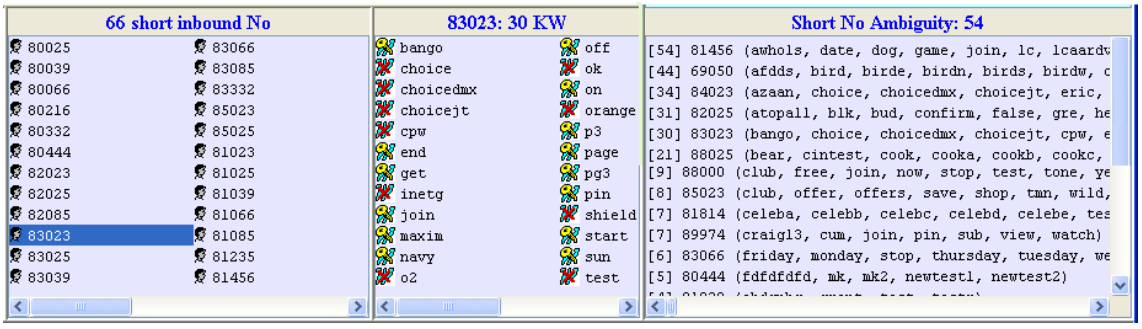

Figure 3. INo ambiguity

To provide such analysis the Knowledge Base (KB) has been created and used for KW, INo and bad MSM analysis, and KW and INo restoration. The main features of KB have been discussed in details in [4]. Here we would like to notice that in our case, under the KB organisation we would understand the regularity of data (INo and KW) distribution in memory assuring the storage of various links between them. At any time KB deals only with relatively *small fragments* of the external world. So, the corresponding structures are needed to integrate these fragments separated in time into the integral picture. The structures obtained as a result of integration should contain more information than it had been used for its creation. The organisation of KB should make allowance for such features as:

- associability;
- ability to reflect similar features for different objects and different features for similar objects (where objects are represented by KW and INo);
- heterarchical organisation of information [5]. The idea of heterarchical approach means that a full association of INo and KW represent very complicated net of nodes and unidirectional links between them. The predetermined hierarchy of "super-" and "subclasses" is absent; every node (INo or KW) is a "patriarch" in its own hierarchy if some process of search initiates with it.

Bad Messages Analysis

The main purpose of Bad Messages (BdM) is to classify BdM and allocate types of BdMs which might be restored. Several hundred thousand BdMs have been detected and result of this is as follows:

- Wrong KW among valid and invalid KW - 42.12%;
- Wrong KW among valid KW - 20.11%;
- Wrong KW among invalid KW - 22.01%;
- Wrong INo - 39.53%;
- "Stop" MSM - 8.78%;
- Empty MSM - 6.47%;
- Wrong alphabet (e.g. Russian) - 2.65%;
- Mobile slang (from T9 dictionary) - 0.37%;
- Rude MSM - 0.08%.

Remark: *Wrong INo* means literally wrong INo, e.g. 22120000, or unknown INo. So despite that 39.53% of *wrong INo* it would not be effective to spend more effort in trying to decrease this percentage. In the next session of paper some ideas of KW and right INo restoration will be discussed.

Algorithm of KW and/or INo Restoration

1 INo recognition. There are four possible type of INo: (i) ***valid***; (ii) ***invalid***, (iii) ***unknown*** when either length of INo is different from short or long INo, or INo does not exist in KB. Remark: Checking existing INo in KB would be sufficient to find out if the INo is known or not. But this operation requires more time than simply checking the length of the INo, and (iv) ***wrong*** INo. Initial analysis of INo does not allow the identification of this type of INo. It would only be possible to do this when KW of the MSM is recognised.

2 Initial MSM validation. MSM will be classified as valid if only contains symbols from the Latin alphabet and/or digits are used. Hereafter, only valid MSM will be considered.

3 Separators elimination from MSM.

4 Fillers elimination from MSM. For example, in MSM: *"I'd like to stop sending messages"* ***I'd like to*** is a filler and will be deleted.

5 Slang elimination from MSM using T9 dictionary.

6 Stop MSM recognition. Remark: In the current version of algorithm MSM *"s_t_o_p"* will not be recognised as a *stop* MSM.

7 Extracting set of KW from KB related to INo, i.e. $\{KW_{INo}\}$, where $\{KW_{INo}\} \subset \{KW_{KB}\}$. $\{KW_{KB}\}$ represents all existing KW in KB.

8 Extracting KW from MSM, i.e. KW_M. Remark: In the current version of the algorithm only the first word of MSM is considered as a KW_M.

9 Extracting set of INo from KB related to KW_M, i.e. $\{INo_{KWm}\}$, where $\{INo_{KWm}\} \subset \{INo_{KB}\}$.

10 Pair INo $\leftrightarrow$ MSM is accepted if $((INo \in \{INo_{KWm}\} \wedge KW_M \in \{KW_{INo}\}) \Rightarrow$ IS-Correct(MSM)) $\mapsto$ return(KW_M), where predicate *IS-Correct(MSM)* is true when *"MSM is correct"* and false (i.e. $\neg$IS-Correct(MSM)) - otherwise. Symbol $\Rightarrow$ stands for word *then* and symbol $\mapsto$ means *lead to*. Returned KW_M is used for further analysis.

11 Pair INo $\leftrightarrow$ MSM represents BdM, if
$(INo \in \{INo_{KWm}\} \wedge KW_M \notin \{KW_{INo}\}) \oplus (INo \notin \{INo_{KWm}\} \wedge KW_M \in \{KW_{INo}\}) \oplus (INo \notin \{INo_{KWm}\} \wedge KW_M \notin \{KW_{INo}\})$,
where symbol $\oplus$ means exclusive or.

12 After recognition of BdM reason, the attempt to restore BdM is undertaken. To explain this step let us assume that the reason of BdM is:
$INo \in \{INo_{KWm}\} \wedge KW_M \notin \{KW_{INo}\}$.
From this it follows that:
$INo \in \{INo_{KWm}\} \wedge KW_M \notin \{KW_{INo}\} \wedge (KW_M \in \{KW_{KB}\} \oplus KW_M \notin \{KW_{KB}\})$.
If $KW_M \in \{KW_{KB}\}$ then attempts to correct INo should be undertaken. The next step will describe the more complicated case of KW_M correction when $KW_M \notin \{KW_{KB}\}$.

13 KW_M correction. There are two different approaches to restore KW_M:

(1) The first approach provides searching $KW_i \in \{KW_{KB}\}$ under several conditions:

- the difference in length of words KW_i and KW_M must be less or equal 1;
- just two different symbols might be in KW_i and KW_M. This rule covers four possible types of misspelling (the word *attempt* is used to demonstrate the first three types): (i) *attepmt*; (ii) *atempt*; (iii) *attembt*, and (iv) *ozlo*. The last type should be considered more attentively. There are two different reasons for this type of misspelling:
  I. Problem of symbol recognition. Very often it is simply impossible for the user to distinguish the letter *'l'* from the digit *'1'*, especially when, for example, the previous symbols are letters but for correct KW digit *'1'* need to be typed in, e.g. *oz10*.
  II. Easier typing. For the user it is easier to press the button 0 once than to press the button 6 three times to enter the letter *'o'* in word *bonus*, because for any reader it is still easy to understand word the *b0nus*. Another example, when instead of the letter *'l'* (pressing the button 5 three times), or *'i'* (pressing the button 4 three times) entered digit *1* e.g. *tab1e*.
- Similarity of words KW_i and KW_M must be more or equal to some Threshold of Similarity (TofS), i.e. $SmIrt(KW_i,KW_M) \geq TofS$. The calculation of $SmIrt(KW_i,KW_M)$ as a percentage is quite simple:

  $$SmIrt(KW_i,KW_M) = (ACS_{LR}(KW_i,KW_M) + ACS_{RL}(KW_i,KW_M)) * 2 / (Length(KW_i) + Length(KW_M))*100,$$

  where $ACS_{LR}(KW_i,KW_M)$ and ACS_{RL} stand for Amount of Compared Symbols from Left to Right and Right to Left respectively. For example, for considered words: *attepmt*; *atempt*; and *attemppt* the values of $SmIrt(KW_i,KW_M)$ are as follows:

  Smlrt(*attempt*,*attepmt*) = (4+1)*2/14*100=71.43%,

  Smlrt(*attempt*,*atepmt*)=(2+4)*2/13*100=92.31%, and

  Smlrt(*attempt*,*attemppt*)= (6+1)*2/15*100=93.33%.

Remark: In the result of comparison of words *attempt* and ***attemppt*** from *left to right* two sequences remain to be compared from *right to left*: *t* and ***pt***. That is why $ACS_{RL}(KW_i,KW_M) = 1$. The compact description of first approach to restore KW_M might be presented in the following manner:

$$\exists\, KW_i\, ((KW_i \in \{KW_{KB}\}) \wedge (SmIrt(KW_i,KW_M) \geq TofS)) \mapsto return(KW_i),$$

where quantifier $\exists$ means *exist*.

To find out an appropriate value for TofS thousands of BdMs have been tested for three different values of TofS – 50.0%, 75.0%, and 100%. The decreasing of restored KWs are:

$$6{,}370 \rightarrow (-1{,}137) \rightarrow 5{,}233 \rightarrow (-709) \rightarrow 4{,}524.$$

That is caused by type 1 of misspelling (wrong sequence of two letters), because $SmIrt(KW_i,KW_M)$ is very sensitive to a word's length, e.g. SmIrt(*node*,*ndoe*)=50.0%, SmIrt(*table*,*tabel*)=60.0%, and SmIrt(*axmpridel*,*amxpridel*)=77.78%. In the current version of the algorithm TofS = 75.0% because type 1 misspelling occurs very seldom in short words (i.e. with a length less than 6 characters).

(2) If the previous approach was not success then algorithm is trying to find such $KW_i \in \{KW_{KB}\}$ that is

(i) an initial part of KW_M, i.e. $KW_i \triangleright KW_M$,

(ii) $\forall KW_i\, (KW_i \in \{KW_{KB}\} \wedge KW_i \triangleright KW_M)$ Select(max(Length(KW_i)), where quantifier $\forall$ means *from all* and Select(max(Length(KW_i)) stands for *"select KW_i with maximum length", and*

(iii) $(Length(KW_M) - Length(KW_i)) \leq (Length(KW_M)/2)$, e.g. *airtext* $\triangleright$ *airtextww3514*.

14 INo correction. Result of KW_M correction is shown on Figure 4. To describe the INo correction let us suppose that pair "*81025* $\leftrightarrow$ *cash*" has been entered. This pair has been recognised as BdM because

$INo \notin \{INo_{KWm}\} \wedge KW_M \notin \{KW_{INo}\} \wedge KW_M \in \{KW_{KB}\} \wedge INo \in \{INo_{KB}\}$. $\{INo_{cash}\}$ = {84025, 86025, 87025, 82085, 87085, 87023}. It would be not acceptable to advise the user: *"Please try to dial 84025, 86025, 87025, 82085, 87085, or 87023"*. Instead a heuristic approach is used and might be describe as follows:

- For each button define a set of *"direct neighbour"* buttons (DrctN) and a set of *"diagonal neighbour"* buttons (DgnlN). Given terms easy to explain by example: DrctN(5) = {2, 4, 6, 8} and DgnlN(5) = {1, 3, 7, 8}.
- Find out the wrongly pressed button. For the considered example, SmIrt(*81025*,*84025*)=80%. The same result that we have for INo *86025* and *87025*. Thus it is very likely that the wrongly pressed button was 1.
- Now the right button should be selected. DrctN(1) = {2, 4} and DgnlN(1) = {5} associated with button 1. First of all the right button is searching among DrctN(1). It is easy to see that only button 4 could be the right button and that is why INo 84025 is displayed (see Figure 5).

Figure 4. KW_M correction

Figure 5. INo correction

The result of testing both KW and INo correction is represented on Figure 6.

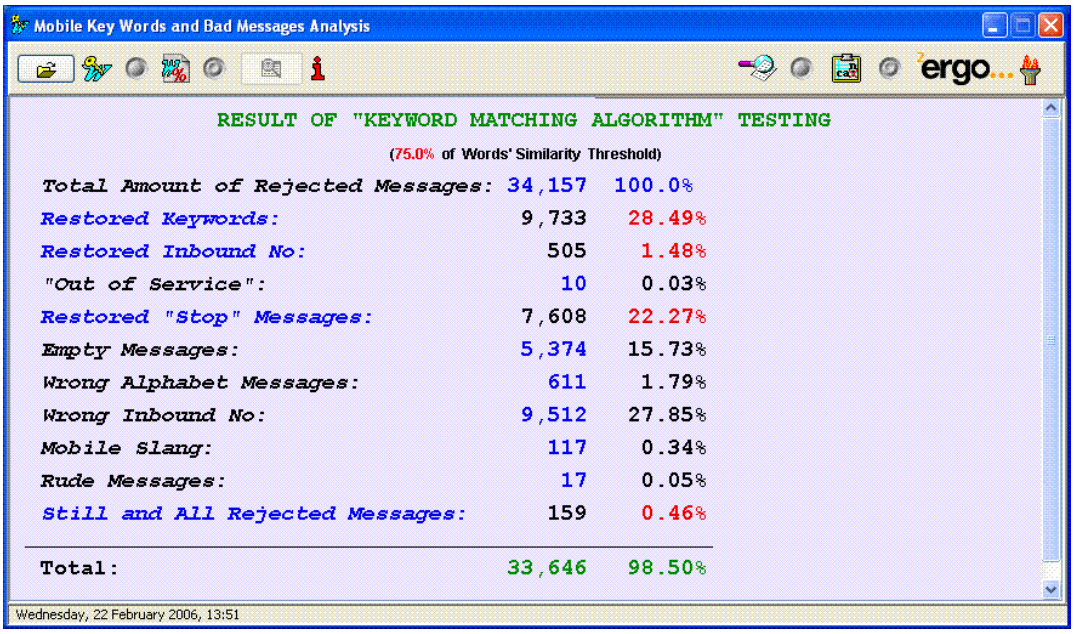

Figure 6. Result of Algorithm Testing

Remark: In Figure 6 the amount of distinct *"Still and All Rejected Messages"* is displayed and that is why the initial amount of BdM = 34,157 is more than the total amount of tested and corrected messages (33,646). The described algorithm improved BdM recognition by 52.25%.

Conclusion

The recent development in natural language processing has made it clear that formerly independent technologies can be harnessed together to an increasing degree in order to form sophisticated and powerful information delivery vehicles. Written speech, verbal speech and MSM analysis provide complementary functionalities, which can be combined to meet the modern technologies requirements.

Bibliography

[1] D.Burns, R.Fallon, P.Lewis, V.Lovitskii, S.Owen, "Verbal Dialogue Versus Written Dialogue*, *Proc. of the XI-th International Joint Conference on Knowledge-Dialogue-Solution: KDS-2005*, Varna (Bulgaria), 336-244, 2005.

[2] Opinion Research Corporation, www.orc.co.uk.

[3] Tegic Communication, www.tegic.com.

[4] V.A.Lovitskii and K.Wittamore, "DANIL: Databases Access using a Natural Interface Language", *Proc. of the International Joint Conference on Knowledge-Dialogue-Solution: KDS-97*, Yalta (Ukraine), 282-288, 1997

[5] M.R.Quillian, "Word concepts: A theory and simulation of some basic semantic capabilities", *C.I.P. working paper 79*, Cornegie Inst. of Technol., Pittsburgh, 1965.

Authors' information

Ken Braithwaite – e-mail: ken.braithwaite@2ergo.com

Mark Lishman – e-mail: mark.lishman@2ergo.com

Vladimir Lovitskii – e-mail: vladimir@2ergo.com

David Traynor – e-mail: david.traynor@2ergo.com

2 Ergo Ltd, St. Mary's Chambers, Haslingden Road, Rawtenstall, Lancashire, BB4 6QX, UK