Developing a Spell Checker as an Expert System

Šandor Dembitz*, Petar Knežević* and Mladen Sokele**

*Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia
**Croatian Telecom, Zagreb, Croatia

In this paper we discuss some basic concepts of knowledge engineering, expressing our feeling that there is something missing. In our opinion, the missing link, able to clarify all the basic concepts, should be the concept of energy, not in use in knowledge engineering practice and theory at all. Since energy is an extremely abstract concept, which led to misunderstandings even in physics, we humbly introduce it by describing an existing expert system, called Hascheck. Hascheck is a spell checker for Croatian language, implemented as learning (semi)automaton, accessible via E-mail and Web, and being in public use nearly 10 years. We also present here a mathematical model of Hascheck's learning. The model led to the so-called cognoelectrical analogy, which allowed some energy considerations about learning, and knowledge.

Keywords: spell checker, knowledge acquisition, modelling of learning, energy.

1. Introduction

Knowledge management is currently attracting a great deal of interest in scientific and business communities. However, is knowledge engineering a scientifically well-founded discipline? There are some reasons to doubt this.

"Expert systems have to be an applied discipline, not a theoretical one. In application, real-word needs such as knowledge-based maintenance and learning need to be accommodated. We need to see expert systems as primarily a discipline to do with supporting knowledge applications; primarily an applied discipline, primarily concerned with the logic of the knowledge and only secondarily a technological discipline. This needs a modification of viewpoint on the part of many expert systems practitioners." (Taylor, 1999; p. 8)

The quoted thoughts of R.M. Taylor bring knowledge engineers in the position of ancient Roman civil engineers constructing bridges. It is not such a bad position. Not knowing Newton axioms, they constructed many bridges, some of which are still in operation, from Spain to the Near East, from England to Africa. But, for modern engineers, this position is very weird; they are taught to think axiomatically. It is not so easy when knowledge is concerned.

On the other hand, there are some attempts to formalise fundamentals of knowledge engineering. How it works will be illustrated by quoting only one of these attempts, (Uschold, 1998). In his introduction M. Uschold says: "Very importantly, *this is a **descriptive** exercise, not a normative one*" (p. 5). A few pages later he says: "In particular, readers are expected to know how the term KNOWLEDGE is being used, and what INFERENCE is. They should be familiar with fundamental notations of LANGUAGE in general and KNOWLEDGE REPRESENTATION LANGUAGES in particular, as well as what is meant by a KNOWLEDGE BASE" (p. 8).

Do we all mean the same when using terms like *knowledge, reasoning* or *language*? Surely not! Speaking of *language*, the authors of these lines primarily think of their own, Croatian language.

Is there any knowledge and reasoning out of language? Of course there is. Artists producing paintings or music have such knowledge and such reasoning. But, is there any sharable, common knowledge and reasoning out of natural languages? We doubt it.

Regarding Uschold's and others' papers about theoretical foundations of knowledge engineering we got a strong feeling that there is something missing, something that could connect, physically, all basic, weakly defined concepts. Being engineers, we believe that we have found the missing link in the concept of energy.

"Knowledge is power", people say. Everyone will agree that learning is difficult. Thus, in everyday life we describe knowledge and knowledge acquisition, or learning, with energy terms. Nowadays many researchers and engineers are trying to build up learning automata, or knowledge-based systems. Do they consider their attempts through energy analysis? Usually not, because of the lack of scientific concepts which could support such considerations.

Even in physics, energy is an extremely abstract concept, which led to many misunderstandings: only *perpetuum mobile* is to be mentioned as an example. Therefore, our paper has to be regarded as a humble – but practical – attempt, pointing where to research further in order to enable energy considerations about knowledge and learning.

At the very beginning we had a simple – for many years now commonly regarded as unattractive – knowledge engineering task: to collect word types and build up an acceptable spell checker for Croatian language. Our approach was unconventional: we decided to implement our checker as a telematic service embedded in E-mail, so the users had to send their texts to a given address and to wait for an automatic reply in the form of the checker's report. Texts sent for checking demonstrated to be a valuable source of new knowledge for the checker. This provoked our interest for learning in technical systems.

The simplicity of our original task had two faces: positive and negative. Firstly the negative: we dealt with common knowledge, correctly and incorrectly written Croatian words. Each naïve user, and most Hascheck users are "naïve" in the sense of the spell checker – expert system – development problem, having some competence reparding Croatian language, could judge whether our product functioned well or not. Fortunately, they accepted Hascheck. The positive side of our job was that we did not have to cope much with structures, relations and similar aspects complicating most knowledge engineering tasks. These circumstances allowed us to go fairly deep in modelling learning. The results will be presented in this paper.

Section 2 describes the checker. Section 3 provides a mathematical model of the learning process based on data collected during the first 5 years of the checker's life; data collected later brought nothing new to the model. Finally, Section 4 introduces a cognoelectrical analogy that allows some energy considerations about knowledge and learning.

2. Hascheck – A Learning (Semi)automaton

"Spelling is one of the best examples I've seen of the need for prototyping: build something small, try it, see how useful it is in practice, then modify and extend" (Bentley, 1985; p. 460). This idea led to Croatian Academic Spelling Checker, called Hascheck (an acronym derived from its full Croatian name: **H**rvatski **a**kademski **s**pelling **check**er). Hascheck is the first Croatian spell checker; some other Croatian checkers were developed later (Sokele, 1997). Hascheck functions as a telematic service embedded in E-mail (the address: `hacheck@fer.hr`). In the last few years it is also accessible via World Wide Web on the address `http://www.hr/hacheck/`. The service has been in operation since March 21, 1994.

Hascheck's initial dictionary counted less than 100,000 common word types. Here we must explain the term word and how we use it. A lemmatised word is what one finds on the left side of a conventional dictionary. By applying its morphology, a natural language produces word forms for each lemmatised word. Not all produced word forms, regarded as alphabetic strings, are necessarily distinct. In English, for example, the noun *work* and the verb *work* are two distinct word forms, because they are two distinct lemmatised words, but one word type. Further, it is necessary to distinguish common words (common nouns, verbs, adjectives etc.) from names (proper nouns, acronyms, foreign words in original orthography etc.), because of their different behaviour in texts. Finally, words in text are tokens. But, not all tokens are words. Some tokens are nonwords (misspellings or typos), and the spell checker's job is to find (detect) them.

With a small initial dictionary we encountered several problems. Croatian is a Slavic language, like Russian, Polish, Bulgarian and others. They all belong to the group of highly inflected languages with a great variety of different word forms – and word types – for each lemmatised word. A Russian estimation sets the lowest limit of dictionary size for an acceptable spell checker at 1,000,000 word types; the upper limit is 100,000,000 word types (Dolgopov, 1986). Having a dictionary 10 times smaller than the estimated lowest limit, we had to find, in order to make the Hascheck report useful for the users, the way to divide unrecognised tokens from a text into legal word types and nonwords, respectively. An idea describing how to treat this problem was found in a paper dating from the very beginning of practical spell checking (Morris&Cherry, 1975).

The AT&T Bell Labs' spell checker *typo* – not in use anywhere for many years now – used the concept of string peculiarity. This concept – despite its original technical implementation – meant that strings like *approachment* and *apprchment* have to be treated differently. For any person having some knowledge of English the first string resembles an English word much more than the second one. Is it possible to build up a program able to make a similar distinction?

An acceptable solution was found by applying binary n-grams (n=3,4,5,6) derived from a common word dictionary. It is a classical AI technique used mostly for error corrections (Riseman&Hanson, 1974; Ullmann, 1977). Finally, a very human-like fuzzy evaluation of strings not contained in the dictionary appeared, classifying them in several classes of peculiarity. The solution proved to be language-dependant. We experimented both in Croatian and English (Dembitz, 1993), but we never managed to put more than 50% of unknown English word types in the class of the lowest peculiarity; the best result in Croatian was around 80%. By all these attempts the rate of typos and misspellings in the same class varied between 10–15% for both languages.

Using our classifying algorithm we were able to bring majority of nonword types at the beginning of Hascheck report; legal Croatian word types not included in the dictionary, were pushed to the end of the report. Such selectivity, taking into account a small dictionary volume, i.e. a poor coverage of incoming texts at the beginning of service life, was of great importance for accepting Hascheck as a useful service.

It is the variety of word endings that makes a language highly inflected. Text sent for checking became a valuable source of new word types unknown to Hascheck. Inspired by (Weischedel et al., 1993) we developed a tagging algorithm for spell checking purposes. The tagging algorithm is applied to collections of Hascheck reports recorded for learning purposes. The tagging for common words is applied on less peculiar strings. Strings beginning with capitals, or consisting of capital letters only are assumed to be potential names and are treated separately. The tagging for names is applied to all classes.

Our word-guessing method succeeds in tagging correctly more than 70% of unknown Croatian word types. These data refer to common words (common nouns, verbs etc.), while efficiency of the algorithm with name types (proper names, acronyms etc.) is around 50%. In both cases the noise – percentage of typos and misspellings in a set of tagged strings is under 10%.

Final result of our research and development was a spell checker functioning as a learning (semi)automaton. The classifier classifies unrecognised tokens from a text as more or less peculiar. The tagger processes the less peculiar. According to capitalisation, the tagger treats potential common words and potential names separately. It produces collections of new word types that should be learned. After minor human supervision and correction (therefrom prefix *semi-* when describing Hascheck) new word types are added to Hascheck name and common word dictionary, respectively. If necessary, only accepted common word types are used for updating the n-gram base.

Hascheck is open to all word processors. The service is fairly quick in responding: For a book of 100,000 tokens, it takes Hascheck less than 1 minute to reply. During the first 5 years of public life Hascheck received over 2,000 texts for processing, or a text corpus amounting to more than 12,000,000 tokens. Actual number of texts processed by Hascheck is over 3,000, which makes the text corpus larger than 35,000,000 tokens. The difference between the first and the second part of Hascheck's life gives an idea about the quality of its service.

As mentioned earlier, Hascheck started to operate with a dictionary of less than 100,000 common word types. In the first 5 years its knowledge grew to more than 300,000 Croatian common word types, and more than 80,000 name types. The increase in the next 5 years (approximately) – this period was not taken into account by modelling that follows – was not so prominent. Only 100,000 new common word types and 50,000 new name types were learned from the corpus of over 20,000,000 tokens.

3. Learning Process

Hascheck learns words to improve its covering of Croatian texts. In order to keep track of learning, an elementary statistical record follows each processing. On the basis of these records, two variables – text coverage (TC) and learning index (LI), are calculated:

$$TC = \left(1 - \frac{NumberOfUnrecognizedTokens}{TextVolume}\right) \cdot 100 \quad (1)$$

$$LI = \left(\frac{NumberOfWordTypesLearned}{TextVolume}\right) \cdot 100 \quad (2)$$

To describe Hascheck learning process over the time it was necessary to find the connection between TC and LI, respectively, and the volume of previously processed corpus (PPC). PPC, the total amount of text processed by Hascheck before the checking of a particular text file, is the only acceptable measure of Hascheck's maturity. PPC is the natural substitution for time in analytical modelling of Hascheck's learning.

We took 20 LI values obtained by proofreading the Croatian Lexicon (Dembitz&Sokele, 1998). The Croatian Lexicon was split into 20 text files. These files came in for checking separately, with long time spans, during the period between November 1994 and May 1997. The Lexicon proofreading was an extremely well controlled process. Therefore, we are sure that statistical data obtained from this process are very reliable.

Using PPC as the time variable, we looked for functions that best fit the data. The number of free parameters in all tested functions was limited to 3 or less. The best fit was obtained by using:

$$LI = a + (100 - a) \cdot e^{-\frac{PPC - \Delta t}{\tau}} \quad (3)$$

Construction of the function (3) needs an explanation. Learning index LI, as defined in (2), cannot exceed the value of 100. The extreme value of LI can be reached only at the very beginning of learning, when the "time", $PPC - \Delta t$, equals 0. The "time" shift Δt is also easy to explain. To put Hascheck in operation, capable to receive the first user's text and to give a satisfactory response, some amount of research, development and programming was needed, as well as processing of a certain amount of text in order to acquire basic Croatian word knowledge. All these are expressed through Δt parameter.

The result of fitting is presented in Fig. 1, together with optimal values of free function parameters a, Δt and τ, and with obtained correlation coefficient r. It is worth mentioning that even physicists experimenting with some fundamental physical law, would be satisfied with the correlation of 0.975.

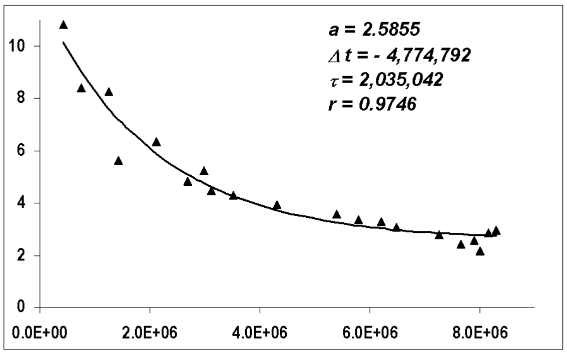

Fig. 1. Fitting of learning index LI $(X - axis = PPC\,[\text{tokens}];\ Y - axis = LI)$.

Independent of learning indexes we took 20 TC values representing average text coverage calculated on the basis of a 3-month period of processing. These 20 points have a total span of 5 years or 12,000,000 tokens, when counted in PPC. By fitting these data we fixed the parameters Δt and τ on values obtained in the previous fitting. The function we chose was a natural choice:

$$TC = b \cdot \left(1 - e^{-\frac{PPC - \Delta t}{\tau}}\right) \quad (4)$$

The result of this fitting is presented in Fig. 2. The fitting was again very good, since the correlation coefficient (parameter r in Fig. 2) was near 0.92.

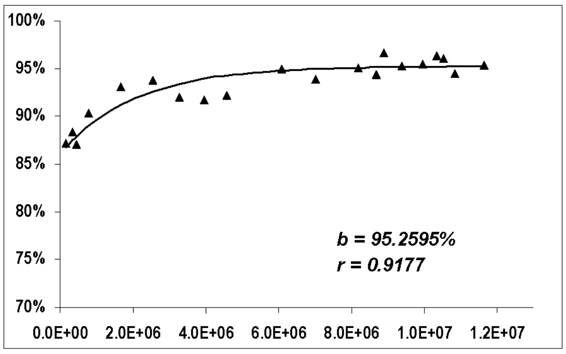

Fig. 2. Fitting of text coverage TC ($X - axis = PPC$ [tokens]; $Y - axis = TC$ [%]).

Functions (3) and (4) correspond to human experience. Learning words is a never-ending process, since every living language constantly produces new words; this is expressed by the small constant a in Fig. 1. "3 to 5% of word tokens are usually missing in the lexicon (*dictionary*) when tagging a real-world text" (Mikheev, 1996). The value obtained for b in Fig. 2 confirms these borders.

This correspondence with experience is giving us the faith that our mathematical model of learning makes sense.

4. Cognoelectrical Analogy

Analogy is not the best way of scientific reasoning. However, if there is no better way of thinking, one must take this one.

Functions very similar to (3) and (4) describe the charging of a real capacitor connected to a real battery (Fig. 3) where $\tau = C \cdot R_C \cdot R_s/(R_C + R_s)$. According to the standard notation in circuit theory, the battery is represented by an electromotive force voltage source with constant voltage E and source resistance R_s. The capacitor is represented by a capacitance C and capacitor resistance R_C. When they are connected, the current (i) and the voltage (u_C) follow the equation (5) and (6), respectively.

$$i = \frac{E}{R_C + R_s} + \frac{E}{R_s} \cdot \frac{R_C}{R_C + R_s} \cdot e^{-\frac{t}{\tau}} \tag{5}$$

$$u_C = \frac{E \cdot R_C}{R_C + R_s} \cdot (1 - e^{-\frac{t}{\tau}}) \tag{6}$$

The power function, $p(t) = i \cdot u_C$, has an extreme if $R_C > R_s$. When the extreme of power exists, it is maximum in $t = \tau \cdot \ln[2R_C/(R_C - R_s)]$.

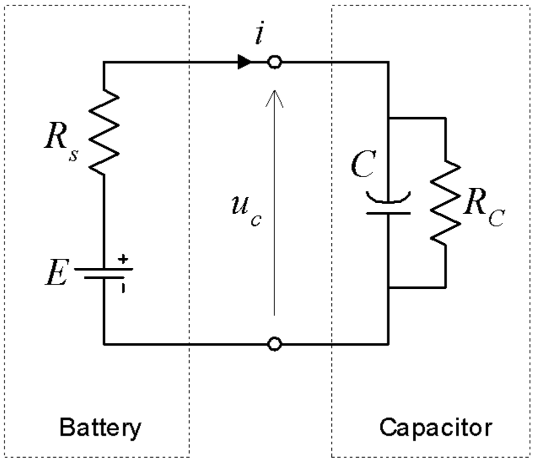

Fig. 3. Electrical equivalence of Hascheck learning process.

Words are the charge for Hascheck. The flow of words, expressed by learning index LI, can be regarded as an equivalent to current i. Analogously, the text coverage can be regarded as voltage u_C. Users, with their natural language competence, are the battery; Hascheck is the capacitor. We call this a *cognoelectrical analogy.*

Having equivalences between LI and i, and between TC and u_C, it is normal to construct and analyse the *power of learning* function (PoL):

$$PoL = \left[a + (100 - a) \cdot e^{-\frac{PPC - \Delta t}{\tau}}\right] \cdot b \cdot \\ \cdot \left(1 - e^{-\frac{PPC - \Delta t}{\tau}}\right) \tag{7}$$

The function (7) has an extreme when the "time" has a value of (parameters are taken from Fig. 1):

$$PPC - \Delta t = \tau \cdot \ln \frac{200 - 2a}{100 - 2a} = 1,465,315\,[\text{tokens}] \qquad (8)$$

Now we must go back to the consideration of the value of "time" shift $\Delta t = -4,774,792$ [tokens], obtained by fitting. As we have already said, this value expresses the fact that Hascheck needed acquisition of basic word knowledge before it could become a useful service. However the "time" shift amounting of nearly 5 Mtokens cannot be verified by the text corpus used for it; only 800,000 real tokens, processed by Hascheck before it was offered to public use, can be quoted (Dembitz, 1993). The rest has to be considered as another type of time in knowledge acquisition, the time needed for research, development and programming of Hascheck.

Therefore we split the entire life of Hascheck in two periods: its virtual time, approximately 2τ long, when Hascheck was only an idea, and its real time, approximately 7τ long, when Hascheck was coping with real texts.

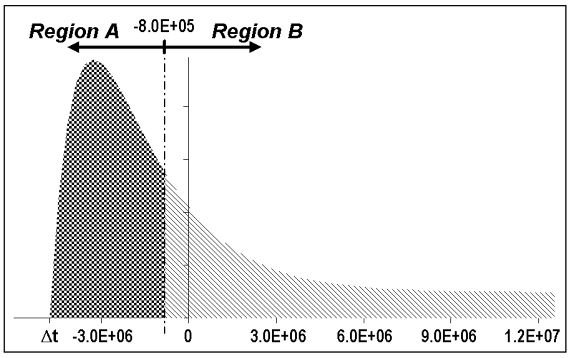

Fig. 4. The Hascheck power of learning function: *Region A = energy spent in virtual time of Hascheck's life; Region B = energy spent in real time of Hascheck's life;* ($X - axis = PPC$ [tokens]; $Y - axis = PoL$).

The Hascheck power of learning function PoL is presented in Fig. 4. It is obvious that the most energy-demanding period was the virtual time of Hascheck's life. The doubts, when one is faced with many problems to be solved, consume much energy. During the real time of Hascheck's life, when we had these problems mainly solved, the demand for energy dropped significantly. Here the analytic model proves our own experience.

Generally speaking, these energy considerations do not contradict common sense. Automata are built up to save human energy; users sent their text to Hascheck to save their own time needed for proofreading. In case of learning automata – or semiautomata, like Hascheck – it should be compensated by the authors' energy spent in the course of making such an automaton operable. To put it simply, learning automata operate due to the *power = knowledge* initially supplied by their authors. Whether they will function well or not, it depends upon the quality and quantity of knowledge initially supplied, and in each concrete case an estimation of these is still pure art.

Since expert systems are still an applied discipline, and not a theoretical one, let us conclude with the last Hascheck practical achievement. In the first week of September 2003 Hascheck checked the entire *Comprehensive Dictionary of Croatian Language*. The 4th edition of Anić's well known dictionary (Anić, 1991, 1994, 1998) is to appear soon under a new title, because of an increase of vocabulary volume covered by the Dictionary. Before its appearance in the book-shops the publisher wanted to check the final lay-out of the Dictionary, so he decided to use Hascheck. The results were astonishing (for the publisher). In the text of 1,300,000 tokens Hascheck detected more than 600 typos and misspellings. Globally, for the publisher, it was a very good rate because 0.05% error-rate is acceptable even in better lexicographies than Croatian, but the people who spent a decade or more working on the Dictionary project couldn't believe they had never noticed these errors. From the Dictionary proofreading Hascheck learned only 2,000 new word types.

5. Conclusion

The results we have presented here are consequences of a well-chosen and very unconventional approach. Being engineers, we always regarded natural language as an economic and energy-balanced system. This point of view is not new. It was very modern in technical literature 50 years ago (Shannon, 1951; Zipf, 1949), but was later replaced by structural and algebra-based approaches to language (Chomsky, 1957). Inspired by Martinet's philosophy of language (Martinet, 1960), we decided to explore the hidden, natural language energy, or geometry of language, in order to solve, with minimal effort, the problem of developing a good spell checker for our highly inflected language. We believe that our achievement, presented here, might produce some impact on the renewal of energy considerations about language and knowledge, since these two concepts are inseparable.

References

[1] ANIĆ, V., *Rječnik hrvatskoga jezika (Dictionary of Croatian Language, 1st edition 1991, 2nd edition 1994, 3rd edition 1998)*, Novi Liber, Zagreb, Croatia, 1991, 1994, 1998.

[2] BENTLEY, J., A Spelling Checker, *Commun. of the ACM*, 1985, 28(5):456–462.

[3] CHOMSKY, N., *Syntactic Structures*, Mouton&Co., The Hague, 1957.

[4] DEMBITZ, S., *Automatic Misspelling Detection and Telecommunication Services* (in Croatian), PhD Thesis, Faculty of Electrical Engineering, University of Zagreb, 1993.

[5] DEMBITZ, S. & SOKELE, M., Computational Proofreading of the Croatian Lexicon, *Proc. of the 9th Mediterranean Electrotechnical Conference*, 1998, pp. 1370–1374, Tel-Aviv, Israel, May 18–20.

[6] DOLGOPOV, A.S., Automatic Spelling Correction, *Cybernetics*, 1986, 22(3):332–339.

[7] MARTINET, A., *Eléments de linguistique générale*, Armand Colin, Paris, 1960.

[8] MIKHEEV, A., Unsupervised Learning of Word-Category Guessing Rules, *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, University of California, Santa Cruz, 1996, pp. 62–70.

[9] MORRIS, R. & CHERRY, L.L., Computer Detection of Typographical Errors, *IEEE Trans. on Professional Communications*, 1975, PC-18(1):54–64.

[10] RISEMAN, E.M. & HANSON, A.R., A Contextual Postprocessing System for Error Correction Using Binary n-Grams, *IEEE Trans. Comput.*, 1974, C-23(5):480–493.

[11] SHANNON, C.E., Prediction and Entropy of Printed English, *Bell System Technical Journal*, 1951, 30(1):50–64.

[12] SOKELE, M., Croatian Spelling Checkers (in Croatian), *WIN.INI*, 1997, 6(3):38–49.

[13] TAYLOR, R.M., Expert Systems in the Knowledge Management Era, *Applications and Innovations in Expert Systems VI* (R.Milne et al., Eds), 1999, pp. 3–8, Springer-Verlag.

[14] ULLMANN, J.R., A Binary n-Gram Technique for Automatic Correction of Substitution, Deletion, Insertion and Reversal Errors in Words, *Computer Journal*, 1977, 20(2):141–147.

[15] USCHOLD, M., Knowledge Level Modelling: Concepts and Terminology, *The Knowledge Engineering Review*, 1998, 13(1):5–29.

[16] WEISCHEDEL, R., METEER, M., SCHWARTZ, R., RAMSHOW, L. & PALMUCCI, J., Coping with Ambiguity and Unknown Words through Probabilistic Models, *Computational Linguistics*, 1993, 19(2):359–383.

[17] ZIPF, G.K., *Human Behavior and the Principle of Least Effort*, Cambridge: Addison-Wesley, 1949.

Received: April, 2002
Revised: November, 2003
Accepted: December, 2003

Contact address:

Šandor Dembitz, Petar Knežević
Faculty of Electrical Engineering and Computing
University of Zagreb
Unska 3
10000 Zagreb
Croatia
Phone: +385 1 6129760
Fax: +385 1 6129616
e-mail: sandor.dembitz@fer.hr
petar.knezevic@fer.hr

Mladen Sokele
HT – Croatian Telecom Inc.
Corporate Strategy and Development Department
Hebrangova 32–34
10000 Zagreb
Croatia
e-mail: mladen.sokele@ht.hr

ŠANDOR DEMBITZ received the BSc (1973), MSc (1981) and PhD (1993) degrees in electrical engineering, all from the University of Zagreb, Croatia. He is presently an assistant professor at the Faculty of Electrical Engineering and Computing, University of Zagreb. His current research interests include natural language processing and expert systems.

PETAR KNEŽEVIĆ received the BSc (1974), MSc (1977) and PhD (1982) degrees in electrical engineering, all from the University of Zagreb, Croatia. He is presently an associate professor at the Faculty of Electrical Engineering and Computing, University of Zagreb. His current research interests include formal description techniques for protocol specification, performance analysis and system modelling.

MLADEN SOKELE received the BSc (1983) and MSc (1987) degrees in electrical engineering, both from the University of Zagreb, Croatia. He is presently at the Corporate Strategy and Development Department of HT (Croatian Telecom), where he runs the Research and Information Centre. His current research interests include economic indicators modelling and forecasting in telecommunications.