Speech Therapy Based on Expert System

Mihai H. ZAHARIA, Florin LEON
Gheorge Asachi Technical University of Iaşi
Bd.-ul D. Mangeron, 53 A, RO-700050, Iaşi
mike@cs.tuiasi.ro

***Abstract*—in this paper the design and implementation of an expert speech therapy system is presented. A brief presentation of involved speech therapy process is made. Also the base for designing and implementations of this system is presented too.** [1]

***Index Terms*—artificial intelligence, expert systems, handicapped aids, medical services, speech treatment**

I. INTRODUCTION

Nowadays in rich societies there are sometime enough human resources specialized in speech disorder therapy of the child or for the adults. Unfortunately even in this case not all persons can afford this type of personalized treatment that involve a human expert who will interact a few hours each day with each child in order to assure him a good feedback in the time of his special home speech exercises given by the speech therapy expert.

As result the computer based either using a classic PC or a mobile device, seems to be suitable. Of course that will not imply the lack of a human therapist in handling child treatment but will ensure a good execution of daily home exercises that are vital for the treatment.

This approach have also many advantages for treating persons from isolated communities by mixing the telepresence of the speech disorder therapist with exercises executed by the patient.

After long analysis with the specialist in speech therapy the conclusion were that a classic approach it is not suitable in design of this kind of systems. The rules are not exactly quantifiable and sometimes may vary on each case.

As result the design of a expert system based applications were elected to permit he specialist to give us in terms of general rules the treatment execution and surveillance.

This system will work only in conjunction with a very good speech recognition therapy that is required to make comparisons between the correct work pronunciation and the child form that need to be corrected.

II. SPEECH THERAPY

The logopedic treatment is in the field of voice pathology. This kind of therapy is very complex and has many stages. The first stage is referred as general therapy and consists in various exercises for specific muscular structures mainly involved in the speech therapy. At this stage a computer can only give a visual representation for the child of the needed to be executed physical exercises.

The second stage is the specific one. Here the child is instructed to repeat a set of words or phrases on well defined time intervals or until an improvement in pronunciation of words is obtained.

As we previous mention the real process controlled by speech specialist is extremely complicated and have many rules that may vary or not for each letter of the alphabet. The next level is at the word pronunciation phase here again a lot of rules and combinations are made and finally the phrase level must be reached.

The quality of voice recognition algorithms is determinant for the use of this approach. All experiments proves that there are no problems at the sound or word pronunciation recognition but at the phrase levels some errors may occur so the impact of computer supervised lessons at this level is diminished.

Any treatment involves the following general procedure. After first exam the child will receive a diagnose and a recommended base therapy strategy; this strategy have a component when the therapist directly work some exercises with the child and then a series of exercises are given as homework to be repeated daily until the following meeting.

When the next meeting occurs the therapist will analyze if there are improvements or not and depending by the result of this examination the homework may be changed or not.

The process can continue indefinitely until a limit of improvements is reached or the speech deficiency is cured.

The expert system can be involved both at the therapist and the patient level.

At the therapist level the expert system can give the following facilities:

1. to create a special treatment log for each child by direct input from therapist or by downloading from a mobile device used to control homework execution the current treatment, ant the results of any homework;
2. to analyses and automatic extract new rules fine tuned for any patient;
3. treatment suggestions based on the personalized profile of child evolution.

At the patient level on the mobile device the local rules from the current homework will be followed by the system.

Here we present a small part of the rules involved in treating the ‘b’ pronunciation defects – betacism that is a part from specific training phase [3]:

1. The therapist/computer will elect a number n of repetitions for the ‚b’ by the child where $n \in [3,10]$;
2. the computer will ask to child to speak lauder the ‚b’;
3. the result is analyzed, compared with a standard pronunciation and then the result is placed in one of the three levels of quality pronunciation determined by the following intervals [0, lim1), [lim1, lim2),

---

[1] This work were supported from CEEX 56 granted by Client National center for programme management - Minister of National Education and Research during the period 2006-2008

Digital Object Identifier 10.4316/AECE.2009.01014

[lim2, lim3), [lim3, 1) where the limits are given by the therapist;

4. Some visual outputs are provided for the child as follows
    a. If the result ∈ [0, lim1), then „Bravo, you have one flower!"
    b. If the result ∈ [lim1, lim2), then „Bravo, you have two flowers!"
    c. If the result ∈ lim2, lim3), then „Bravo, you have three flowers!"
    d. If the result ∈ [0, lim1), then „Bravo, we can proceed to the next step"
5. If the quality of child pronunciation ∈ [0, lim3), then repeat from step 3 and decrement n;
6. If the quality of child pronunciation ∈ [lim3, l] then goto COMP procedure
7. If the result ∈ [0, lim2) then the child must go back to general procedure exercises
8. if the result ∈ [lim2, lim3) then goto 1
9. if ∈ [lim3, l] then goto COMP procedure

III. EXPERT SYSTEMS PROBLEMS

The activity and the presence of a human expert in establishing a model for solving a class o similar problems is required because the human have some knowledge about some patterns about the designed model, his performances, the way of combining with or within other models and also about the restrictions in use[2]. A informatic system based on knowledge is a system that can model with a good precision the abilities of the human expert in solving problems of a specific domain of activity [4].

CLIPS (C Language Integrated Production System) is an expert system shell developed by the Software Technology Branch, NASA. It is thus specifically designed to facilitate the development of software to model human knowledge or expertise [7].

There are three ways of representing knowledge in CLIPS: rules (which are primarily intended for heuristic knowledge based on experience), generic functions (which are primarily intended for procedural knowledge), and objects-oriented programming (also intended for procedural knowledge, supporting classes, message-handlers, abstraction, encapsulation, inheritance, and polymorphism).

The CLIPS shell provides the basic elements of an expert system: the fact list and instance list, which represent the global memory for data, the knowledge base or the rule base that contains all the rules, and an inference engine that controls the overall execution of rules, based on an implementation of the Rete algorithm [6]. Using the inference engine, rules match patterns on objects and facts.

A naïve implementation of an expert system might check each rule against the known facts in the Knowledge base, firing that rule if necessary, then moving on to the next rule (and looping back to the first rule when finished). For even moderate sized rules and facts knowledge-bases, this naïve approach performs far too slowly.

The Rete algorithm provides the basis for a more efficient implementation of an expert system. A Rete-based expert system builds a network of nodes, where each node (except the root) corresponds to a pattern occurring in the left-hand-side (the condition part) of a rule. The path from the root node to a leaf node defines a complete rule left-hand-side. Each node has a memory of facts which satisfy that pattern.

As new facts are asserted or modified, they propagate along the network, causing nodes to be annotated when that fact matches that pattern. When a fact or combination of facts causes all of the patterns for a given rule to be satisfied, a leaf node is reached and the corresponding rule is triggered.

The Rete algorithm is designed to sacrifice memory for increased speed. In most cases, the speed increase over naïve implementations is several orders of magnitude, because Rete performance is theoretically independent of the number of rules in the system [8].

CLIPS use the forward chaining reasoning procedure. In this case, only a small fraction of the rules in the knowledge base are actually triggered by the addition of a given fact. Therefore, a great effort is make to construct partial matches repeatedly that have some unsatisfied premises. A solution would be store and gradually complete the partial matches as new facts arrive.

The Rete algorithm preprocesses the set of rules in the knowledge base to construct a dataflow network in which each node is a literal from a rule premise. Variable bindings flow through the network and are filtered out when they fail to match a literal. If two literals in a rule share a variable, then the bindings from each literal are filtered through an equality node. A variable binding reaching a node for an n-ary literal might have to wait for bindings for the other variables to be established before the process can continue. At any given point, the state of a Rete network captures all the partial matches of the rules, avoiding many recomputations [1].

IV. EXPERT SYSTEM "LOGO EXPERT"

The developed expert system has, as we expected to implement various sets of different exercises but most of them have many similitudes.

From basic Use Case analysis result that many different windows to be generated for each exercise. As result the primary design phase were restarted.

Now using intersections based on exercises similitude we obtain 9 generic types for exercises that can be particularized for each of 34 exercises needed for any alphabet sound.

To store the exercises texts given by the speech therapist a *.RTF file were elected in order to preserve the Romanian language special characters.

At the second level of application design we choose to use the Factory design pattern due to inherent similitudes for the exercises. The use is depicted as follows:

```
while (!end)
 {
      DecisionForm f = null;
      int exerciseType = ExerciseType(currentState);
      switch (exerciseType)
      {
       case 1: f = new SimpleDecisionForm(currentState); break;
       case 2: f = new ComplexDecision1Form(currentState);break;
          ...
       case 9: f = new Logigram4Form(currentState); break;
       default: return;
       }
   }
```

Each main class associated to a specific exercise type has a helper class. In associated object for helper class are stored the exercises texts, the dynamic selections made by the therapist, the number of repetition and also the quality of standard pronunciation that is compared with the child input.

Of course here the standard pronunciation records given by the therapeutic may be replaced by the use of a cepstrum based speech synthesis library [5]. This approach may give a boost in performance of phrases recognition but also it is possible to decrease de efficacy of the treatment.

In the figure 1 and decision schema for decision making only for betacism treatment is presented.

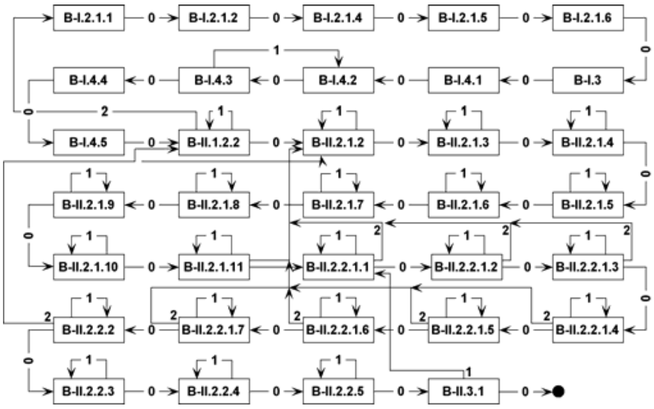

Figure 1. State machine given by expert system decision for betacism treatment.

The *StartForm* class contains a *ClipsWrapper* object that is an bridge to CLIPS inference engine that is implemented in one C native code made DLL. There is no special problem in working with unmanaged code at this level.

In the following are presented a part of the expert system rules used for betacism treatment are presented.

```
(defrule rule-B0
        (decision -1)
        =>
        (printout "end" crlf)
)
...
(defrule rule-B10
        (state B-I.4.3)
        (decision 0)
        =>
        (printout t "B-I.4.4" crlf)
)
(defrule rule-B11
        (state B-I.4.3)
        (decision 1)
        =>
        (printout t "B-I.4.2" crlf)
)
...

(defrule rule-B15
        (state B-II.1.2.2)
        (decision 0)
        =>
        (printout t "B-II.2.1.2" crlf)
        (printout t "Reset" crlf)
)
(defrule rule-B16
        (state B-II.1.2.2)
        (decision 1)
        =>
        (printout t "B-II.1.2.2" crlf)
)
```

In the base class DecisionForm we also have function to set the number of re-executing an exercise between one to maximum three times base don automat decision which is made using the experience provided by results on previous exercises or direct by the input of the speech therapist.

The deployment structure for the system is presented in figure 2.

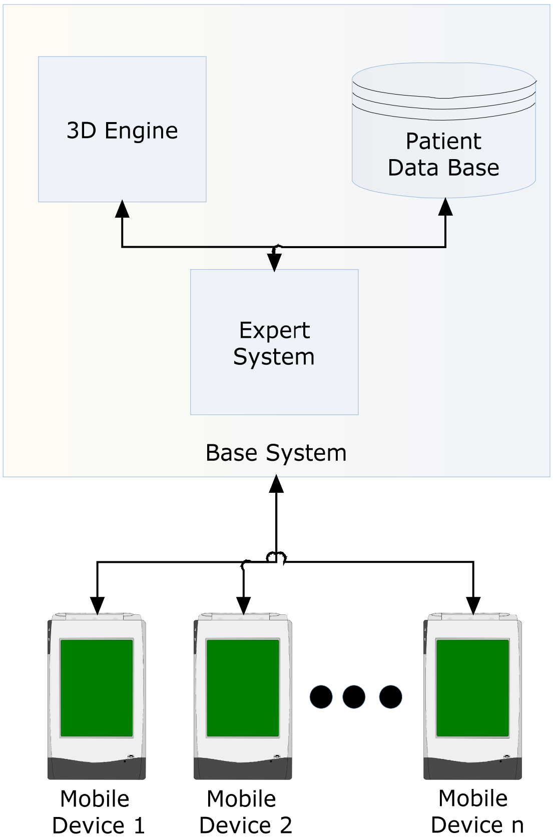

Figure 2. Speech Therapy system.

One of the functions that uses the CLIPS inference engine is depicted as follows:

```
private void MakeDecisionWithClips(ref string state, int decision) {
            clips.Assert("(state " + state + ")");
            clips.Assert("(decision " + decision + ")");
            string result = clips.Run();
            string[ ] lines = result.Split("\r\n".ToCharArray());

            if (lines == null || lines.Length == 0) {
                state = "end";
                return;  }
            if (lines[0] == "Mid-process decision") {
                // DECISION ON THE FIINAL OF
                //STAGE II.2.1 OF TREATMENT
                int step = state.IndexOf('-');
                string therapyType = state.Substring(0, step);
             if(DecisionForm.II_2_1_Total- DecisionForm.II_2_1_Wrong
                >= 0.8 * DecisionForm.II_2_1_Total) {
                    state = therapyType + "-II.2.2.1.1";
                    DecisionForm.ResetAllRuns(); }
                else {
                    state = therapyType + "-II.2.1.2";
                    DecisionForm.ResetAllRuns(); }     }
            else {
                state = lines[0];
                if (lines.Length > 3) {
                    if (lines[3] == "Reset")
                           DecisionForm.ResetAllRuns(); }     }
```

V. SYSTEM EXTENSION

Distributed artificial intelligence is at the intersection between artificial intelligence and distributed computing. It is considered to be “the study and design of systems formed of many interacting entities, distributed logically and often spatially, that can be considered in a certain sense autonomous and intelligent” [9]. Most of the classical artificial intelligence systems are static, and their architecture is predefined, while agent-based systems are dynamically modified in time.

The use of agents is motivated because they are a solution for complex systems management. Also, the recent development of technology and especially the Internet requires the possibility that different applications communicate and interact. In general, for heterogeneous information environments, that are geographically distributed and complex in size, a centralized approach is practically impossible. As result the mobile computing can be use to make distribute computing using agents.

These open environments forget some time the fact that the security and control requirements must reflect the value of handled information and also the potential loses that can appear if the security policy is a failure or not exist [10].

Agents work in open environments and need to be autonomous, heterogeneous and updated or modified dynamically. Agents are becoming a popular topic for research and development in applied functions, since they provide a natural means of performing the previously mentioned tasks in uncontrollable and dynamic environments. They can be constructed locally to the requirements of the user group.

The agents will make possible knowledge exchanges with other similar systems. A list of licensed location will be downloaded from a central server and then the workstations will directly make exchanges of knowledge by the use of intelligent agents.

An agent that will work onto this platform will be at one precise time moment in only one of the following states:

1. New – it has just arrived into the current platform or it is a new borne agent.
2. Running – at this moment it executes something according to his own rules for execution and communication
3. Suspended – appears when the agent waits for an event to come. In fact the multithreading approach will ensure us that the agent will be fully suspended only if he must depart from the current platform. Otherwise some threads will be suspended and other will continue to work.
4. Dead – the agent finish its execution

The complexity of this approach requires that the agent has some supplementary states that are related to interaction with the framework. The agent can be also in the unregistered or registered state. In the first case, the agent cannot exchange information with other colleagues. Before a transfer is made, the two involved platforms must establish a negotiation protocol in order to see if that is suitable. One situation can be if the targeted platform is overloaded but the transfer needed to be executed than this will be delayed. In order to communicate, the agents will use the KQML (Knowledge Query and Manipulation Language) [11]. The frameworks will exchange information by the use of some common XML like files that provide an asynchronous communication type which is necessary to avoid some temporary overloads.

At this level of privacy which is high because confidential data about the patient are involved. So it is better to develop or use an agent framework like JADE that will consult the existent list of local rules and will make information exchange with other similar servers from the Internet [12]. The advantage is that even this server is broken the access at the patient database is very difficult. The database will store the personalized information about any patient into a secured manner [13].

VI. CONCLUSION

The presented system is designed into a modular fashion. This allows us to be very flexible from life cycle development point of view. The use of the inference engine simplify the software architecture and give the possibility of learning continuous new rules from the expert selections analysis taken on each personalized case. As result the accumulated knowledge can be easily exchanged with similar systems in order to quick increase the aide of the speech therapist using the new added experiences.

ACKNOWLEDGEMENTS

The research reported in this paper was supported by the national research grant Research of Excellence, TERAPERS, CEEX 56/2006.

REFERENCES

[1] S. J. Russell, J. Stuart, P. Norvig, “Artificial Intelligence: A Modern Approach”, Englewood Cliffs, NJ, Prentice Hall, 1995.

[2] A. Florea, St.Gh. Pentiuc, D. Kayser, “ Intelligence artificielle et agents intelligents” Ed.Printech Bucarest, Romania,2004.

[3] I. Tobolcea et al, Research report at CEEX 56 research grant, 2007.

[4] C.S. Krishnamoorthy, S. Rajeev,” Artificial Intelligence and Expert Systems for Engineers”, CRC Press, 1996

[5] M.Tatham, K. Morton, “Expression in Speech:Analysis and Synthesis”, Oxford University Press, 2006.

[6] C. Forgy “Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem”, Artificial Intelligence, vol. 19, pp. 17-37,1982.

[7] J. C. Giarratano, “CLIPS User’s Guide”, www.ghg.net/clips/ download/documentation/ usrguide.pdf, 2002.

[8] Wikipedia, The Free Encyclopedia, “Rete algorithm”, en.wikipedia.org/ wiki/Rete_algorithm, 2008.

[9] G. Weiß, S. Sen, ”Adaptation and Learning in Multiagent Systems”, Berlin: Springer Verlag, 1996.

[10] C. Iglesias, M. Garijo, J. Gonza’lez, “A survey of agent-oriented methodologies”, in Intelligent Agents V. Agents Theories, Architectures, and Languages, Lecture Notes, in Computer Science, vol. 1555, eds. J. P. Mu¨ller, M. P. Singh, and A. S. Rao, Springer-Verlag, 1998.

[11] M. Berthold, J. D. Hand, “Intelligent Data Analysis – An introduction”, Springer, 1999.

[12] M. Cerlinca, A. Graur, Ş. G. Pentiuc - Simulation Of Switch Box Routing In Fpga, Advances In Electrical And Computer Engineering, Suceava, Romania, ISSN 1582-7445, No 1/2002, volume 2 (9), pp. 86-90

[13] N. T. Melita, I. Popescu, S. Holban, A Genetic Algorithm Approach to DNA Microarrays Analysis of Pancreatic Cancer, Advances in Electrical and Computer Engineering, Suceava, Romania, ISSN 1582-7445, No 2/2008, volume 8 (15), pp. 43-48.