DECISION MAKING ALGORITHM FOR DISPATCHING PATIENTS IN DISASTER SITUATIONS

Cosmin TOADER[1], Nirvana POPESCU[2]

Abstract - This article proposes an algorithm to dispatch patients to the best suited hospitals according to the patient's severity in case of a natural disaster or a mass accident. In the context in which multi-agent systems have been quickly adapted for the e-health area, the proposed solution uses a previous work related to patient flow control based on multi-agent systems. Thus, a decision-making algorithm will be presented and studied both with Multi-agent system's support and without. The proposed algorithm will compute a priority for each scenario, deciding where the patient will be dispatched.

Keywords: multi-agent systems, e-health, patient dispatch.

1. Introduction

E-health problems have been getting an increased attention over the last years. However, disaster situations like earthquakes, fires, mass-accidents have not been treated in many researches. An algorithm for fast dispatching patients to the relevant hospitals with the relevant free beds and available personnel is proposed, that will be proven more efficient due to a multi-agent system use, based on a previous work [1].

In a previous work we have presented a multi-agent system with the purpose of controlling the patient's flow, from the disaster until stabilized in a hospital [1]. The presented system respected the national sorting protocol for emergency situations of Romania, labeling patience with five severity codes: red, yellow, green, blue and white. It is well known that any hospital's emergency section has three types of rooms: major emergency room, minor emergency room and the monitor room. The system needed to assign the patient to one of the rooms in one of the hospitals, if available. Each room takes care of patients of a certain severity code. However, if there are no rooms available for a patient with a certain code, the system is looking for the next viable option.

Negotiation techniques were presented between the patient agents and the theory of agents working together and making compromises with the purpose of dispatching every single patient to the best fitting emergency room, while respecting the national sorting protocol for emergencies.

[1] PhD student, University POLITEHNICA of Bucharest, Romania, e-mail: cosmintg@gmail.com
[2] Prof., University POLITEHNICA of Bucharest, Romania, e-mail: nirvana.popescu@upb.ro

In this paper an algorithm is proposed, to show how an agent can decide which is the best fitted hospital for a patient and why, as well as what arguments to use in a negotiation with another agent. Once the algorithm is described, both scenarios with and without multi-agent system's negotiation phase will be presented to prove the efficiency.

2. Related work

There are several papers on the e-health topic, yet few of them take multi-agent systems into consideration and fewer even treat disaster situation. A notable paper in this area is "Smart algorithms for patient assignment in disasters" [2], where a smart solution of dispatching patients is proposed. The algorithm proposed by Zubari *et al.* assigns patients from the early stage of a disaster to the nearest available hospital, taking in consideration the trauma rating of the patient, the distance to the hospital and the capacity of the hospital. The trauma rating of the patient is computed by parameters retrieved from several sensors installed at the disaster's location on the patient, including temperature, blood pressure, SpO2 saturation levels and pulse rate. The capacity of the hospital is computed based on the total number of physicians and the number of patients a physician can take care for. Evaluating parameters like distance, trauma rating and the hospital's capacity, a priority is assigned, based on which the patient is assigned to a hospital. Although the algorithm proved more efficient than manual dispatching patients, there is still room for improvement.

Another notable work is the decision support for ambulance dispatch and relocation proposed by Andersson *et al.* in [3], which tries to reduce the waiting time of an ambulance dispatching patients. An ambulance dispatch algorithm is described, as well as a dynamic ambulance relocation method. The purpose of the paper is to optimize the preparedness by relocating ambulances based on the zone's notoriety of calls. This way, whenever a call is made by a patient, an ambulance will always be in proximity. The same problem was approached differently in [6] where a two-tiered ambulance dispatch and redeployment solution was developed considering patient severity classification errors and using an approximate dynamic programming (ADP) algorithm.

Roughly the same idea was found in the emergency medical services priority dispatch proposed in [4], in early years of the research on smart healthcare dispatch services, although this paper is trying to solve the problem of medical experts being sent for BLS (basic life support) cases, and not having enough personnel for actual ALS (advanced life support) cases. The system spared ALS units from 40.2% incidents, making them available for more serious cases.

3. Factors to consider

Based on the national sorting protocol, patients are assigned five code levels:

- Red – patients in highest risk
- Yellow – patients in high risk
- Green – patients in stable condition, but requiring 2 resources to be allocated to maintain the stable vital signs
- Blue – non-urgent patience, only require one resource
- White – patients that do not require immediate assistance

Earlier work [9] has shown how sensors can compute the severity score of a patient on a scale of 1 to 100. Therefore, we can form a coefficient "S" which represents the severity of the patient, S >= 0 and S <= 1.

Based on the severity score, distance to the hospital and available room, an algorithm can be created to assign a priority. However, unlike other work that has been done, we also know that a hospital has multiple emergency room types, each destined for one or more code levels, according to the national sorting protocol for emergencies:

- Major emergencies (resuscitation room)
  - Code red patients
  - Unstable code yellow patients
- Monitor room
  - Code yellow patients
  - Code green patients with complications
- Minor emergencies
  - Code green patients
  - Code blue patients
  - Code white patients

The capacity of the hospital can no longer be computed as other article suggests, since each room has a different weight, a different number of physicians, a different capacity.

So far, the capacity of a hospital was computed by the simple formula:

$$C_H = \sum_{i=1}^{p} C_i \quad (1)$$

$C_H = Capacity\ of\ a\ hospital$
$C_i = Capacity\ taken\ by\ 1\ pacient$

Taking in consideration the three major types of emergency rooms in a hospital, the formula becomes:

$$H = \{ M, Mo, m \} \quad (2)$$

$$C_H = \sum_{i=1}^{p_M} C_i + \sum_{i=1}^{p_{Mo}} C_i + \sum_{i=1}^{p_m} C_i \quad (3)$$

$pM = maximum\ number\ of\ patients\ in\ the\ Major\ emergency\ room$
$pMo = maximum\ number\ of\ patients\ in\ the\ Monitor\ room$
$pm = maximum\ number\ of\ patients\ in\ the\ Minor\ emergency\ room$

And it is still not accurate, since each room can hold patients of certain code levels. Therefore, a weight is proposed to be applied for each emergency room type.

Since the sensors described in a previous article [9] indicate a severity score "S", a sub-unitary number, we can consider the following severity-code mapping:

- $Red = 0.8 - 1$
- $Yellow = 0.6 - 0.8$
- $Green = 0.4 - .06$
- $Blue = 0.2 - 0.4$
- $White = 0 - 0.2$

The weights for each room type are the following:

- Major emergency (M) = 11.1
- Monitor room (Mo) = 1.1
- Minor emergencies (m) = 0.1

They are assigned in such a way so one available spot in the major emergency room will result in a higher priority than all the available spots in a monitor room. The same for one spot in the monitor room being a higher priority than all the spots available in the minor emergencies room. This way, if a patient has a code red assigned, a hospital with only one spot left in a major emergency room will be preferred versus a hospital with zero spots in a major emergency room and 100% spots available in the monitor room

4. Computing the priority

A priority is required to decide the best option for a patient when choosing a hospital. To compute this, we need to consider the severity of the patient, since an unstable patient with code red ($S \geq 0.8$) requires a major emergency room, the distance to the hospital, as well as the available spots in each of the emergency rooms. The distance is computed by the following formula:

$$D = 1 - \frac{D_h}{D_{max}} \quad (4)$$

Where $D_{(max)}$ represents the distance between the patient and the furthest hospital. By this, we will obtain a coefficient between 0 and 1, the higher the coefficient, the closest the hospital is to the patient. The unit of measure for the distance does not matter for the algorithm, since only the coefficient will be used. For the purpose of the experiment, we used seconds.

Next, we need to compute the availability for each of the hospital's room, applying the previously mentioned weight.

$$\begin{cases} \mathrm{C_M} = \frac{\mathrm{C_{Mcurrent}}}{\mathrm{C_{Minitial}}} * 11.1 \\ C_{Mo} = \frac{C_{Mocurrent}}{C_{Moinitial}} * 1.1 \\ \mathrm{C_m} = \frac{\mathrm{C_{mcurrent}}}{\mathrm{C_{minitial}}} * 0.1 \end{cases} \quad (5)$$

(The proposed weights could be any range of numbers, distant enough to make the score difference between the rooms. Powers of 2 would also work (2, 4, 8))

After the first stage of results, with a simulation of 120 patients, the error rate was too high. The reason was the random distribution of the patients' severity. We measure the error by the number of misplaced patients in each of the emergency room. For example, if a high severity patient (code red) ends up in a minor emergencies room, we consider it an error, unless every other room type of every other hospital is filled with high severity patients.

To mitigate those errors, a dynamic weight coefficient was used. That means, we use the weight based on the patient's severity. If the patient has a high severity (code red), then naturally, the major emergency room will have the highest weight. If the patient has a low severity (code blue), then the major emergency room will have the lowest weight.

The final priority is given by the formula:

$$\mathrm{Priority(h)} = \mathrm{SS} + \left(1 - \frac{\mathrm{D_h}}{\mathrm{D_{max}}}\right) + \frac{\mathrm{C_{Mcurrent}}}{\mathrm{C_{Minitial}}} * \mathrm{weight(SS, M)} + \frac{\mathrm{C_{Mocurrent}}}{\mathrm{C_{Moinitial}}} * \mathrm{weight(SS, Mo)} + \frac{\mathrm{C_{mcurrent}}}{\mathrm{C_{minitial}}} * \mathrm{weight(SS, m)} \quad (6)$$

$SS = severity\ score$
$\mathrm{C_{Mcurrent}} = current\ capacity\ of\ the\ Major\ emergency\ room$
$\mathrm{C_{Minitial}} = initial\ capacity\ of\ the\ Major\ emergency\ room$
$\mathrm{C_{Mocurrent}} = current\ capacity\ of\ the\ Monitor\ room$
$\mathrm{C_{Moinitial}} = initial\ capacity\ of\ the\ Monitor\ room$
$\mathrm{C_{mcurrent}} = current\ capacity\ of\ the\ Minor\ emergency\ room$
$\mathrm{C_{minitial}} = initial\ capacity\ of\ the\ Minor\ emergency\ room$

5. Experimental results. Simulation of assigning patients

An initial experiment consisted of the simulation of the following case: random data was generated for 120 patients (random trauma levels and distances), and 4 hospitals were used, holding 30 spots each (10 for each of the hospital rooms: major emergencies room, minor emergencies room and the monitor room).

Naturally, having exactly 120 patients of different trauma levels, they will not match exactly to the 120 spots available across the 4 hospitals, according to the national sorting protocol. The aim of the algorithm is to place them as accurate as possible, without leaving anybody behind. The error rate was measured for each hospital and each room type. We considered an error when a patient of code red, for example, is placed in a minor emergencies room, or when a patient of code blue or green is placed in a major emergencies room. The initial simulation did not involve multi-agent systems, it was based purely on the algorithm described above. The main role of the algorithm is to compute the priority of each patient for each of the available hospitals.

The *ComputePriority()* method calculates the priority of a patient to each of the available hospitals. To do so, a part of the described algorithm is applied:

```
function ComputePriority(patient, hospitals)
{
        SS = patient.Severity;
        Dmax = Max(patient.Distances);

        for each (hospital in hospitals)
        {
                dScore = 1 – GetDistance(patient, hospital) / Dmax;
                CM = (CM_Capacity(hospital) / CM_InitialCapacity(hospital) *
                        CM_Coef;
                CMo = (CMo_Capacity(hospital) / CMo_InitialCapacity(hospital) *
                        CMo_Coef;
                Cm = (Cm_Capacity(hospital) / Cm_InitialCapacity(hospital) *
                        Cm_Coef;
                priority = SS + dScore + CM + CMo + Cm;
                priorities.Add(priority);
        }
        return priorities;
}
```

The coefficients CM_Coef (major emergencies room coefficient), CMo_Coef (monitor room coefficient) and Cm_Coef (minor emergencies room coefficient) are dynamic and computed based on the patient's trauma score:

```
function SetCoeficients(severity)
{
```

```
roomType = GetRoomType(severity);
switch (roomType)
{
        case Major:
                CM_Coef = HIGH;
                CMo_Coef = MED;
                Cm_Coef = LOW;
                break;
        }
        case Monitor:
                CM_Coef = MED;
                CMo_Coef = HIGH;
                Cm_Coef = LOW;
                break;
                }
                case Minor:
                        CM_Coef = LOW;
                        CMo_Coef = MED;
                        Cm_Coef = HIGH;
                        break;
                        }
}
```

The *GetRoomType(severity)* method returns the type of the room the patient needs to be assigned according to the national sorting protocol. The coefficients are assigned based on it. For example, if a patient's trauma score is 99 (code red), then the CM_Coef will have a high value (major emergencies room), because the patient needs to get in the major emergencies room. If the room is fully occupied and no other hospital has available spots in the major emergencies room, the next one should be considered, so the patient needs to get to the monitor room. After establishing each hospital's priority for the respective patient, it needs to be assigned to one of the available hospitals. The assignment algorithm is a simpler one, since the highest priority will always win:

```
function AssignPatient(patient)
{
        priorities = OrderDesc(priorities);
        for each (priority in priorities)
                {
                        hospital = GetHospital(priority);
                        if (AllocatePatient(patient, hospital) == true)
                                {
                                assignedHospital = hospital;
                                break;
                                }
                }
        return assignedHospital;
```

}

where the *AssignPatient(patient, hospital)* method tries to allocate the patient to the respective hospital and returns true or false based on its success status.

The next experiment has been done simulating the following scenario: the same set of data consisting of 100 hospitals and 3000 patients of random severity and distances; a random distribution of patients to the hospitals. Each patient assignment to a hospital was performed based on the available slots for the patient in the emergency room best fitted for the patient's condition. For comparison, the presented algorithm ran over the same set of data.

Our conclusion after these experiments was that the algorithm's results were superior to the random algorithm, resulting in great delivery time. However, error rates were also comparable. An error represents a patient misplaced, e.g. a patient with a low severity placed in a major emergency room, or a patient of high severity placed in a minor emergency room. While both algorithms prefer placing a lower severity patient in a major emergency room rather than placing a high severity patient in a minor emergency room, the errors were still present. The data is forged in such a way that there is no perfect distribution of patients. The results in terms of errors of both algorithms were similar.

6. Experimental results using the multi-agent approach

The next step of the experiment was to involve the multi-agent system already developed by us and presented in a previous paper [9]. Patient-centered multi-agent systems involved in health care has been studied and applied in the recent years [8], [11-13]. The research done so far presents the efficiency of a multi-agent-based system for hospitals by the utilization of agent characteristics and covers only situations when the patient is represented by an agent whose goal is to arrive at the nearest hospital that has a free bed with an available physician, without treating the problem when two patients of different severity are applying for the same place in the same hospital. Our solution has as main goal to reduce the patient assignment error, while keeping an advantage still on the delivery time of the patient.

Involving multi-agent system to help distributing patient might decrease the delivery time of patients, but the assignment error should decrease significantly. As our previous work states [1][9], an agent's self-interest is to get the patient within its responsibility to the closest hospital with an available spot based on the patient's severity (Fig.1). Detecting heart-rate increases and constantly having a low body temperature, could signal a combined pattern with a higher warn level. The system allows multiple pattern configurations creating an agent for each pattern. In this way, a pattern is created based on other patterns,

and each agent will work to collectively contribute to that complex pattern, forming a "pyramid" of agents (Fig. 2).

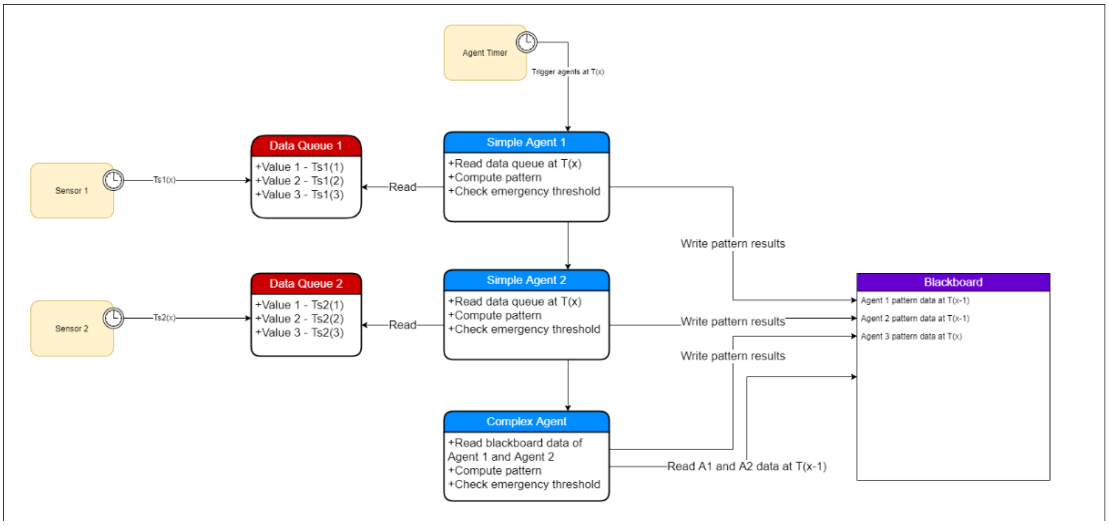

Fig.1. Agent combination example for measuring patient's severity. At every T(x) we have a heart-rate pattern from T(x-1) and a temperature pattern from T(x-1) [9].

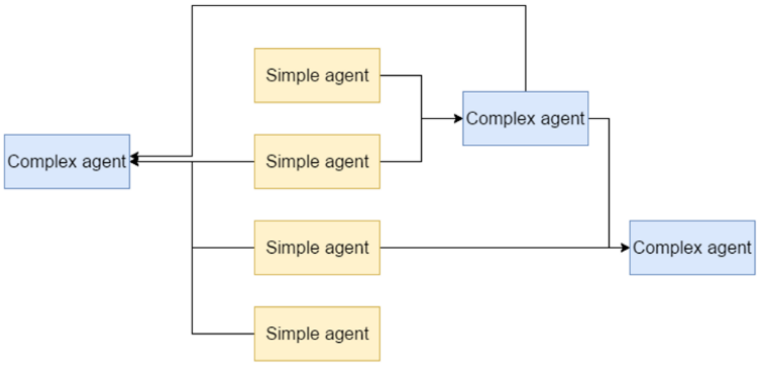

Fig. 2. Complex periodic agent architecture [9]

This, however, has drawbacks on a larger scale. While a critical patient should get the closest spot possible, a self-interested agent would normally also get a green label patient in the closest hospital in the monitor room, if that's the only spot available. To mitigate this, a negotiation technique has been introduced [1]. An agent with a red code patient will not only try to find the closest available major emergencies room but will also try to negotiate with other agents to obtain it, if the room is not yet occupied, but it is booked. That means, the other party (the other agent) will also decide if it's wise for its patient to yield the booked spot and go to the second-best option. This is done by an argument we call a negotiation score.

After querying the same data using the presented algorithm and the multi-agent system, the results are really improving. The patient delivery time shows no performance loss (Fig. 3), and the patient assignment error (Fig. 4) has drastically been reduced, which shows improvement.

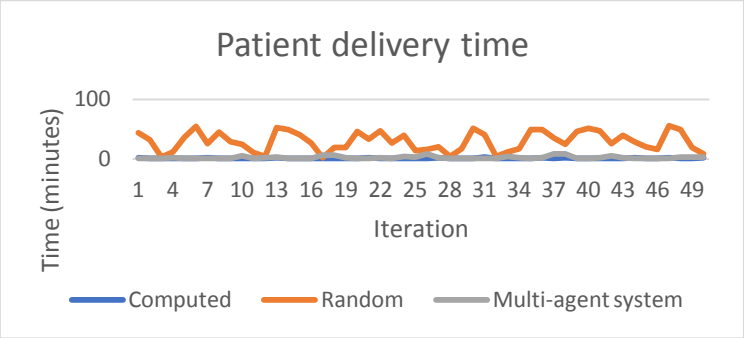

Fig. 3. Patient delivery time in the case of the proposed algorithm (computed) vs random distribution vs multi-agent system

The delivery time overall is somewhat increased, yet the difference is not that high. The assignment error, however, has a bigger improvement.

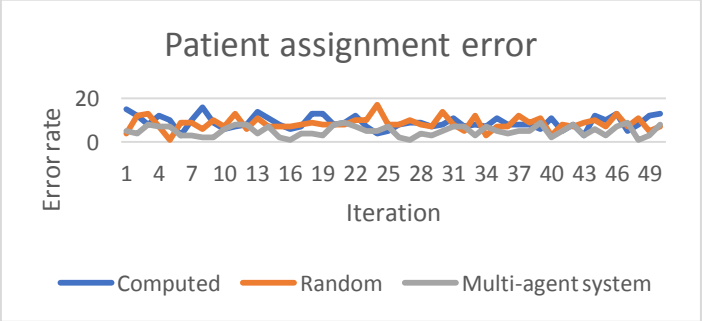

Fig. 4. Patient assignment error in the case of the proposed algorithm (computed) vs random distribution vs multi-agent system

6. Conclusions

In the context in which e-health problems have been getting an increased attention over the last years, the patient-centered multi-agent systems involved in healthcare have been studied and developed offering robust solutions in many cases. The paper proposed an algorithm that has shown how an agent can decide which is the best fitted hospital for a patient and why, as well as what arguments to use in a negotiation with another agent. Both scenarios with and without multi-

agent system's negotiation phase were presented and discussed to prove the efficiency of the solution.

The proposed algorithm did not have a significant effect on the assignment error rate of the patients, at least not a visible improvement. However, it did minimize the delivery time significantly. It does show improvement over other works in the field, though an accurate comparison cannot be done since the scenario is quite different. There are multiple variables that must be considered, and the national sorting protocol [10] has to be respected.

Combining the algorithm for distributing patients with the power of multi-agent systems, the results are improving considerably. We are confident on saying that the algorithm, alongside multi-agent systems can minimize the risk of losing patients in a disaster situation. The solution is expected to behave very well in a real-life scenario. However, there is very little evidence to support the effect of the prioritization of emergency ambulances on patient outcome [5].

Improvements can still be made, as the current simulations did not consider the amount of time for an emergency room to be occupied. An estimation of its occupancy could be done using machine learning techniques to further improve the results.

R E F E R E N C E S

[1] *C. Toader, N. Popescu, I. A. Teodorescu, S. Busnatu, A. D. Toader*, "Patient flow control using Multi-agent systems", in Proc. of 22nd IEEE International Conference on Control Systems and Computer Science (CSCS 22), DOI: 10.1109/CSCS.2019.00047, 2019, pp. 244-250.

[2] *J. A. Zubairi, S. Idwan*, "Smart algorithms for patient assignment in disasters", in ICT Express, vol. 4, issue 2, June 2017, pp. 107-111.

[3] *T. Andersson, P. Varbrand*, "Decision support for ambulance dispatch and relocation", in Operational Research for Emergency Planning in Healthcare, Springer, vol. 1, 2016, pp. 36-51.

[4] *P. A. Curka, P. E. Pepe, V. F. Ginger, R. C. Sherrard, M. V. Ivy, B. S. Zachariah*, "Emergency medical services priority dispatch" in Ann. Emergency Medicine, vol. 22, 1993, pp. 1668-1695.

[5] *P. Hinchey, B. Myers, J. Zalkin, R. Lewis, D. Garner*, "Low acuity EMS dispatch criteria can reliably identify patients without high-acuity illness or injury", in Prehospital Emergency Care Journal, vol.11, issue 1, 2007.

[6] *S. H. Park, Y. H. Lee*, "Two-tiered ambulance dispatch and redeployment considering patient severity classification errors", in Journal of Healthcare Engineering, https://doi.org/10.1155/2019/6031789 , 2019.

[7] *K. Bohm, L. Kurland*, "The accuracy of medical dispatch - a systematic review", in Scandinavian Journal of Trauma, Resuscitation, and Emergency Medicine, 2018.

[8] *F. Bergenti, A. Poggi*, "Multi-agent systems for e-health: Recent projects and initiatives", https://www.researchgate.net/publication/261985355 (accessed on January 2020), 2015.

[9] *C. Toader, N. Popescu, V. Ciobanu*, "Multi-Agent Solution for a Cloud-based e-Health Application", in Proc. of the 22nd IEEE International Conference on System Theory, Control and Computing (ICSTCC), 2018, pp. 683 – 690.

[10] Romanian national protocol for emergency –

https://www.smlugoj.ro/organizare/compartimente/data_files/content/files/protocol-national-de-triaj-2017.pdf (accessed on March 2020)

[11] *N. Benhajji, D. Roy, D. Anciaux*, "Patient-centered multi-agent system for health care", in IFAC - PapersOnLine, vol. 48, issue 3, 2015, pp. 710-714.

[12] *M. K. Moghaddam, M. Shojafar, M. R. Nami, H. Rashidi*, "An Efficient Multi-Agent System for E-health Functionalities", in Int. Journal of Software Enginering and its Applications, vol. 7, issue 3, 2013, pp 24-34.

[13] *A. A. Bielskis, V. Denisovas, D. Drungilas, G. Gricius, O. Ramasauskas*, "Modelling of Intelligent Multi-Agent based E-health Care System for People with Movement Disabilities", in Elektronika ir Elektrotechnika, vol. 6, issue 86, 2008, pp. 37-42.