# Alternative A(H1N1) Suspects Management

DAN ADRIAN MARIOR, RADU ZGLIMBEA, CONSTANTIN CÂRCIUMARU
Department of Automation and Department of Computer Science
University of Craiova
B-dul Decebal, nr. 107, 200440, Craiova
ROMANIA
marior@automation.ucv.ro http://www.ace.ucv.ro
radu@automation.ucv.ro http://www.ace.ucv.ro
ccircium@ford.com http://www.ford.com

*Abstract:* This paper deals with the differential diagnostics process behind the verdict of AH1N1 infection or not for the swine flu suspects. The application was designed as a support for doctors was built with ASP.NET 3.5 technology, the broad spectrum development platform, and it enabled us to create a web application that could be accessible from anywhere on the internet, assist the doctor in the diagnostics process, filter out false positives, manage the patients, and also generate sets of statistical charts for case evolution analysis.

*Key-Words:* A(H1N1), application, diagnostic, management, charts

## 1 Introduction

During the past year we have seen an outbreak of swine flu in many countries of the world, and in Romania those who are suspected of the disease can put their hope for reliable diagnostic and treatment in the hands of the capable doctors from the "Matei Balş Infectious Diseases Institute" in Bucharest.

Transmission of the swine origin influenza virus is thought to occur in the same way as seasonal influenza. Human-to-human transmission occurs by inhalation of large infectious droplets as well as by direct contact with secretions or aerosols. At present, there is no evidence of spread of infection by eating pork, or through water, thus the psychological effect is greater than reality and the name of the disease is not so accurate [1].

In Romania the pandemic psychological impact effect has been quite large, and family doctors could not efficiently manage it, so it was and still is up to the "Matei Balş Infectious Diseases Institute" to impose appropriate measures and reliably diagnose and keep statistics of the persons suspected of contracting the virus. The false positives (those who contracted the common influenza or some other infection) are very hard to detect and filter out, as the symptoms are similar (fever, chills, nausea, vomiting, body aches, lethargy, and fatigue, which usually appear in rapid succession). As it has already been noted from practice, the new flu can lead to death by respiratory failure and other causes like sepsis.

## 2 Problem Formulation

The doctors entrusted with the diagnostics and treatment of AH1N1 influenza in Bucharest are overwhelmed with the number of pacients accusing the symptoms mentioned earlier in the paper, thus an application that can help the doctors organize, filter out the unnecessary and drawing statistical and management conclusions would be most helpful.

Management is in general a complex problem, but in this situation, given the fact that it was a national health problem, the application would have to be fairly reliable, easy to use, highly available and dependable, the requirements of all critical applications.

Security measures had to be imposed by design, as no other person besides the doctors and the administrator of the application should have the right to access the resources the application was supposed to store and process. Programmatically, this can be done by specific permissions on files and folders, or user based access, without restrictions of the operating system the application runs on. Further in the paper we indicate how these security measures were imposed in this case.

## 3 Problem Solution

The ASP.NET 3.5 platform was chosen in order to offer a solution to the inherent problem, being the most appropriate for the facilities it offers, such as independence from hardware architecture (the application runs in a browser on the client machine, this being the only compulsory requirement to run it), very good database connectivity (Active Data Objects, ADO.NET), membership and role management, personalization, site navigation, themes, master pages and other advantages; a positive aspect of implementing such a solution is that the application can be accessed via a browser from anywhere in the world, through the Internet, and if security measures are correctly imposed, the great advantage over the alternative solutions is clear.[2]

Other positive aspects of the ASP.NET choice are: reduced amount of necessary code for writing large applications, Windows built in security and per-application configuration (making applications safer and more secure), better performance due to early binding, just-in-time compilation, native optimization, and cacheing services, language independence, simplicity, pure server side technology, process monitoring and others.

The architecture for the application was naturally chosen to be client/server. Client/server architectures divide applications into two or more components. The client portion uses the functions of the server; in most cases, separate hardware systems are used for clients and servers. "The distribution of the application load across several computers linked in a network keeps the individual units relatively favourable" [3].

For reasons of scalability and optimal distribution of computing resources, the three layer resources distribution approach (business logic layer, database layer and presentation layer) has prevailed as a viable foundation for companies of all dimensions; the former phrase is summarizing the revolution that made central mainframe computing abilities available to more organizations, no matter how wealthy.

Before going into details about the implementation of the client/server architecture in this case, we shall describe certain aspects of this paradigm. The term client/server computing is used i nvarious ways. At least two perspectives must be distinguished to understand this term: the hardware-oriented view and the software-oriented view. The operating systems of the 1980's (for example Microsoft Net and Novell Netware) were an implementation of the client/server paradigm. In those days the idea of client/server computing was thoroughly computer systems oriented, for example LAN-connected desktops to specially designed computers (file servers, print servers, etc.). From this perspective general usage of this idea of computing designates desktop systems as clients conneted to specially designed servers. Restricting the term client/server omputing to certain hardware configurations is not appropriate, as this narrow interpreteation limits the ability to take maximum advantage of the benefits of this architecture.[2]

When we consider it as a combined software and hardware architecture, client server computing reaches its full potential. The concept beneath it is clarified if the modern programming languages structure is considered. They draw the line between main programs and procedures and/or functions.

Subroutines provide certain services to the main program. At runtime the main program calls subroutines and waits for them to end execution before it continues with its own steps in the flow. These are named synchronous subroutine calls. Subroutines themselves can call other subroutines for certain tasks they need done.

The principle subroutines use for calling is expanded in the client/server approach to include both programs that operate on different computers and asynchronous calls. The caling program is called the *master* or client, and the called program is called the *slave*, or the server. In principle, the client and server programs can be installed on a single computer, or they can be installed on different computers, linked by the corresponding communication protocols. During these asynchronous calls the program who calls does not need to wait for the called program to end, but continues immediately with the processing; this manner of processing is applicable for systems connected through a WAN (Wide Area Network) when we cannot assume that the communication links will be available without interruption. The software-oriented view of client/server computing is becoming increasingly important when we want to sepparate between modern business application software and monolithic mainframe computers. For example COBOL which is widespread in the mainframe domain has no implementation of block structuring. In contrast, client/server applications, by being modularized, are characterized by a master/slave relationship betweeen the individual software modules; typically there are modules for graphical presentation, others for application functions and others for data storage. Modern business applications also run background services, for example for background processing and message exchange.

When we consider it a software architecture, client/server computing forms the "basis for cooperative processing of entirely different software conponents and can be implemented either centralized or highly distributed, netowrked installations with a multitude of different servers"[2].

Client/server systems can be set up in various forms, but there are mainly 5 configurations that companies have used in time:

- Centralized system;
- Distributed presentation;
- Database access across computer boundaries;
- Three layer client/server systems with distributed presentation, distributed application logic, and database access logic across computer boundaries;
- Multilayer client/server systems with cooperative processing.

When a centralized system configuration is chosen, presentation logic, application logic and a database management system are installed on the same computer. [4]

In distributed presentation configurations, the processing associated with the graphical display of the user interface is shifted to sepparate presentation computers close to the user. Application logic and database storage run both on a central server.

În database access across computer boundaries configurations we have separate servers for the database and the DBMS (Database Management System). Application logic and presentation are running together on separate computers. There is a lot of communication between the servers and it is usually done by RemoteSQL.

When we compare the client/server architectures discussed so far, three layer systems offer clearly improved flexibility in load distribution. Special computers are implemented spearately for presentation, application logic and DBMS, which allows for a homogenous lload distribution on individual servers with clearly improved response times. Relatively inexpensive computers can be implemented as application servers, since they do not need graphical screens and have none or little need for hard disk storage capacity. [5]

The greatest flexibility is provided by multilayer client/server systems with distribution of the different application and system services on the servers best suited for their assigned role. Communication between the servers is based on synchronous program to program communication, asynchronous message exchange and Remote SQL.

The variant of client/server architecture implemented in the application is the first, with all three servers aggregated in one machine, which makes for a very cost effective sollution, in case a more powerful computer is provided; there is no need to specifiy minimal or optimal system requirements because they are, from any given point of view, above average. The average performance of the three applications heavily dependends, in this case, upon the hardware resources of the assigned system so we can safely say the more powerful the computer (the weaker the bottlenecks are) the better.

In modern systems it seems hard disk drives are becoming more and more of a bottleneck as they are mostly mechanic components of an architecture. Of course, many improvements have been made over the last years and now it is appropriate to say that high performance is no longer out of reach even for the individual persons, to say nothing of companies seeking to wisely invest in computers in order to provide better services for customers, whether they are patients or not.

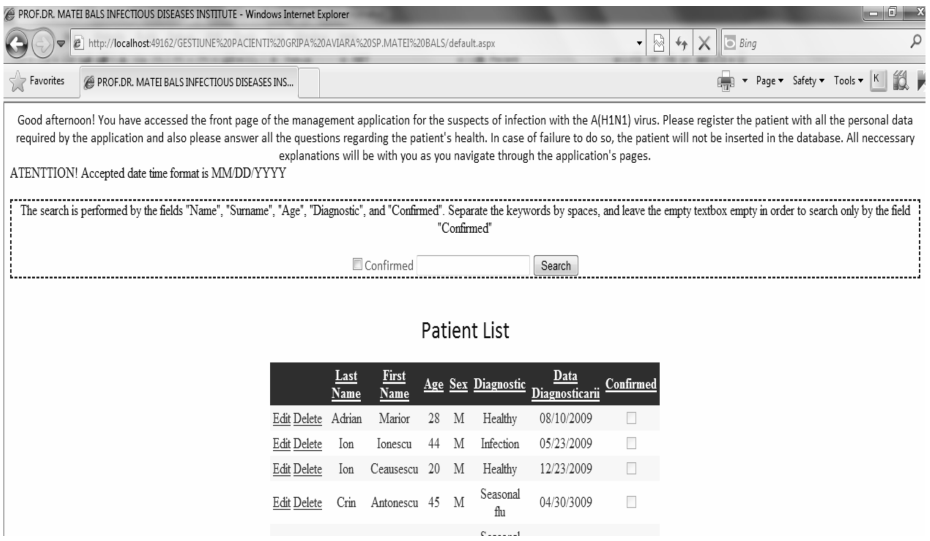
Given the fact that our application is from the web category (by running it in the browser we ensure at least for the application itself there is no need for high end computers) the performance requirements seem to be laid more upon the database server (RAM memory and harddisk transfer speed and latency) and less on the graphical subsystem of the computer.

We continue the paper with some data sets and their corresponding results, from which a person with the right statistical and management knowledge could draw conclusions to optimize the activities.

For the first dataset we have the results shown in figures 2 and 3, with their corresponding input dataset given in table 1, and for the others there is a similar association.

As for security only a few of the problems that could arise have been addressed. For example by enforcing usernames or passwords we made sure none of the other members of the personnel except doctors and the application administrator(s) have acces to the database, which in these cases contains highly sensitive and personal data. In this case, if an attacker or person trying to gain access to the system were to steal the information therein, a possible scenario of consequent actions would be to find out whether a person (public or not) has or does not have A(H1N1) flu, which is not necessarily a tragedy, but in the case of companies which store customer credit card and also personal data losing information could be devastating.

Considering these issues in the future the application could be made immune to cross-site scripting (XSS) which is a common vulnerability found in web servers and web applications alike. Cross site scripting attacks can lead, by using a specially crafted query string, to gaining database access in a matter of seconds. Other commonly exploited flaws are buffer overflows, cross site request forgering, directory traversal, Xpath injection, SQL injection and others.


PROF.DR. MATEI BALS INFECTIOUS DISEASES INSTITUTE - Windows Internet Explorer
http://localhost:49162/GESTIUNE%20PACIENTI%20GRIPA%20AVIARA%20SP.MATEI%20BALS/default.aspx
Bing
Favorites
PROF.DR. MATEI BALS INFECTIOUS DISEASES INS...
Page
Safety
Tools

Good afternoon! You have accessed the front page of the management application for the suspects of infection with the A(H1N1) virus. Please register the patient with all the personal data required by the application and also please answer all the questions regarding the patient's health. In case of failure to do so, the patient will not be inserted in the database. All neccessary explanations will be with you as you navigate through the application's pages.

ATENTTION! Accepted date time format is MM/DD/YYYY

The search is performed by the fields "Name", "Surname", "Age", "Diagnostic", and "Confirmed". Separate the keywords by spaces, and leave the empty textbox empty in order to search only by the field "Confirmed"

☐ Confirmed [ ] Search

Patient List

| | Last Name | First Name | Age | Sex | Diagnostic | Data Diagnosticarii | Confirmed |
|---|---|---|---|---|---|---|---|
| Edit Delete | Adrian | Marior | 28 | M | Healthy | 08/10/2009 | ☐ |
| Edit Delete | Ion | Ionescu | 44 | M | Infection | 05/23/2009 | ☐ |
| Edit Delete | Ion | Ceausescu | 20 | M | Healthy | 12/23/2009 | ☐ |
| Edit Delete | Crin | Antonescu | 45 | M | Seasonal flu | 04/30/3009 | ☐ |

**Fig. 1 Main application page (after login)**

| Name | Surname | Age | Sex | Diagnostic | Diag. date | Conf. ? |
|---|---|---|---|---|---|---|
| Adrian | Marior | 28 | M | Healthy | 08/09/2009 | NO |
| Ion | Ionescu | 44 | M | Infection | 05/23/2009 | NO |
| Alin | Teodorescu | 34 | M | AH1N1 | 09/10/2009 | YES |
| Ion | Ceausescu | 20 | M | Healthy | 12/23/2009 | NO |
| Crin | Antonescu | 45 | M | Seas. flu | 04/30/2009 | NO |
| Nina | Iliescu | 90 | F | AH1N1 | 09/20/2009 | YES |
| Ramona | Badescu | 36 | F | Seas. flu | 03/16/2009 | NO |
| Ionel | Teodoreanu | 20 | M | Bact. inf. | 09/28/2009 | NO |
| Camelia | Partescu | 56 | F | AH1N1 | 08/30/2009 | YES |
| Elena | Tincuț | 67 | F | Seas. flu | 07/07/2009 | NO |
| Marius | Cercel | 44 | M | AH1N1 | 04/05/2009 | NO |
| Andra | Ionescu | 50 | F | AH1N1 | 02/15/2009 | NO |

**Table 1 First dataset**

| Name | Surname | Age | Sex | Diagnostic | Diag. date | Conf. ? |
|---|---|---|---|---|---|---|
| Dan | Ionescu | 59 | M | AH1N1 | 01/12/2009 | YES |
| Corina | Danielescu | 18 | F | Seas. flu | 02/13/2009 | NO |
| Ionel | Danciulescu | 31 | M | Healthy | 02/14/2009 | NO |
| Ionel | Iorga | 29 | M | Healthy | 03/04/2009 | NO |
| George | Becali | 56 | F | AH1N1 | 04/15/2009 | YES |
| Ramona | Theodorescu | 34 | F | Pulm. Inf. | 01/12/2009 | NO |
| Radu | Zglimbea | 28 | M | Healthy | 06/15/2009 | N O |
| Marius | Marian | 30 | M | Healthy | 06/16/2009 | N O |
| Alina | Matei | 34 | F | AH1N1 | 07/17/2009 | NO |
| Daniela | Matei | 27 | F | Healthy | 08/17/2009 | NO |
| Traian | Basescu | 100 | M | Sifilis | 09/18/2009 | NO |
| Emil | Boc | 90 | M | Sifilis | 09/18/2009 | NO |
| Elena | Udrea | 36 | F | Sifilis | 10/19/2009 | NO |
| Robert | Fundeanu | 40 | M | Healthy | 12/20/2009 | NO |

**Table 2 Second dataset**

| Name | Surname | Age | Sex | Diagnostic | Diag. date | Conf. ? |
|---|---|---|---|---|---|---|
| Tim | Roth | 46 | M | Healthy | 01/03/2009 | NO |
| Cicerone | Pascu | 13 | M | AH1N1 | 01/03/2009 | YES |
| Romina | Lascu | 17 | F | AH1N1 | 01/05/2009 | NO |
| Fane | Spoitoru | 50 | M | Pulm. inf. | 02/06/2009 | NO |
| Domnica | Geangu | 78 | F | Healthy | 02/06/2009 | NO |
| Ioana | Teodoreanu | 56 | F | Healthy | 02/07/2009 | NO |
| Florin | Marin | 45 | M | Healthy | 03/07/2009 | NO |
| Nicu | Marinescu | 24 | M | Healthy | 03/08/2009 | NO |

**Table 3 The 3rd dataset**

| Name | Surname | Age | Sex | Diagnostic | Diag. date | Conf. ? |
|---|---|---|---|---|---|---|
| Cerasela | Micu | 34 | F | AH1N1 | 01/02/2009 | YES |
| Daniela | Micu | 45 | F | Healthy | 02/03/2009 | NO |
| Lavinia | Badulescu | 19 | F | Urinary inf. | 02/04/2009 | NO |
| Bogdan | Dragnea | 56 | M | Healthy | 03/02/2009 | NO |
| Costin | Miron | 46 | M | Seas. flu | 04/15/2009 | NO |
| Cornel | Naidin | 70 | M | AH1N1 | 04/30/2009 | NO |
| Alina | Popescu | 33 | F | Healthy | 05/24/2009 | NO |
| Florin | Dracea | 24 | M | Seas. flu | 05/16/2009 | NO |
| Maria | Businescu | 25 | F | Seas. flu | 06/19/2009 | NO |
| Corina | Pentelescu | 56 | F | AH1N1 | 06/09/2009 | YES |
| Daniela | Menchenie | 34 | F | AH1N1 | 06/08/2009 | YES |
| Raluca | Sandulescu | 50 | F | Seas. flu | 07/07/2009 | NO |
| Radu | Matei | 27 | M | AH1N1 | 07/28/2009 | YES |
| Coralia | Marcu | 26 | F | Seas. flu | 07/29/2009 | NO |
| Gabriel | Munteanu | 28 | M | Seas. flu | 07/30/2009 | NO |
| Dorian | Demetriu | 40 | M | AH1N1 | 09/08/2009 | YES |
| Despina | Vandu | 15 | F | Seas. flu | 10/08/2009 | NO |
| Virgil | Miron | 59 | M | AH1N1 | 11/08/2009 | YES |

**Table 4. The 4th and last dataset**

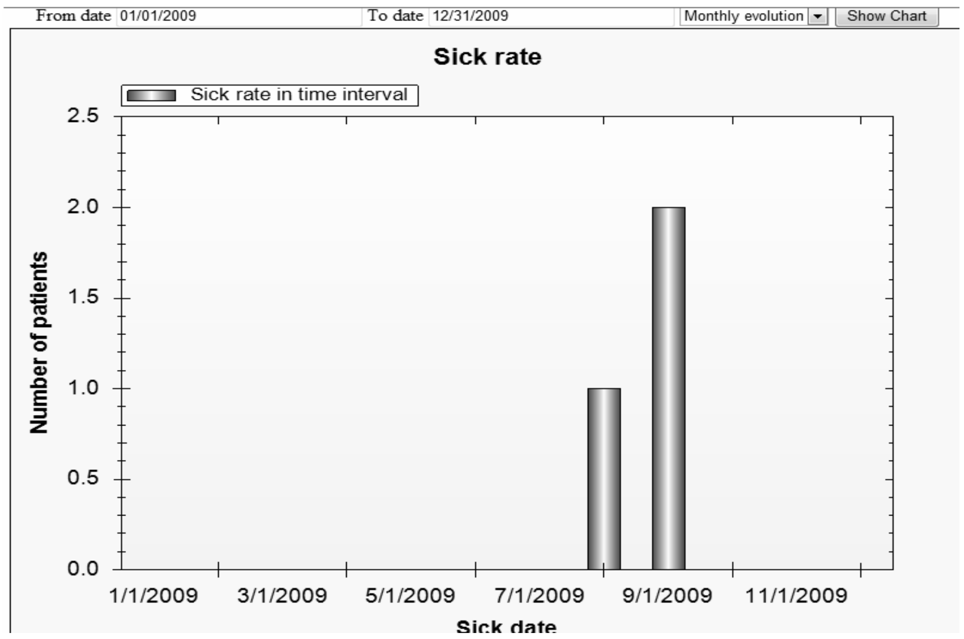

From date 01/01/2009 To date 12/31/2009 Monthly evolution Show Chart

Fig. 2 Monthly evolution 1st dataset

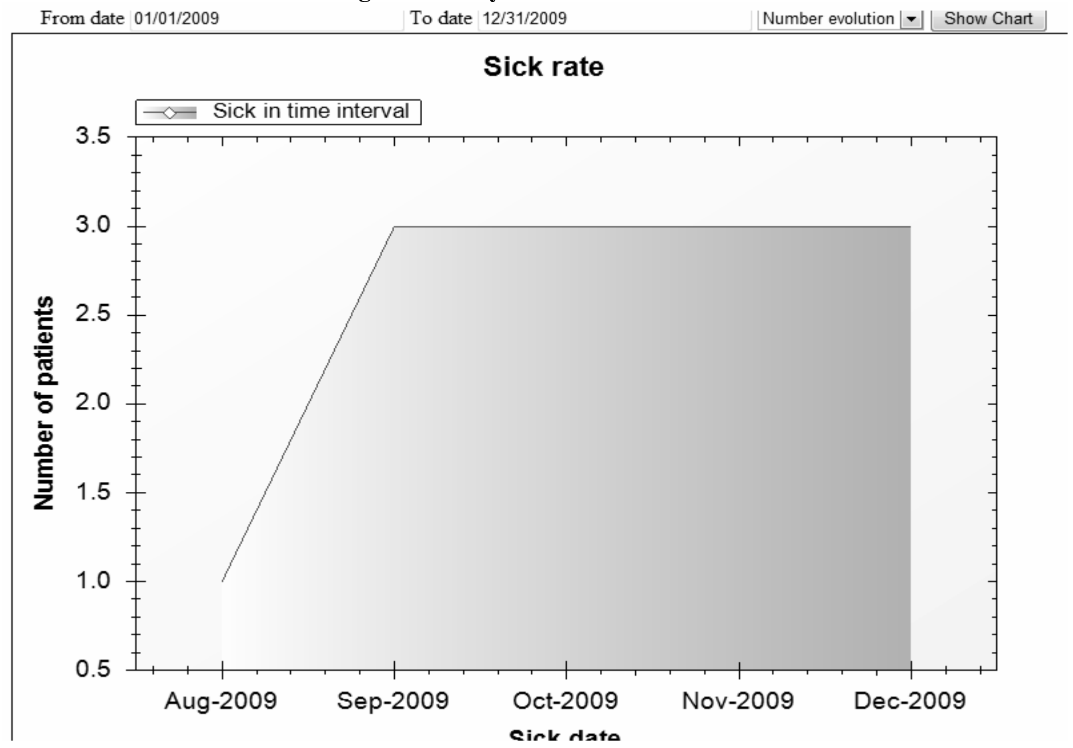

From date 01/01/2009 To date 12/31/2009 Number evolution Show Chart

Fig. 3 Patient number evolution 1st dataset

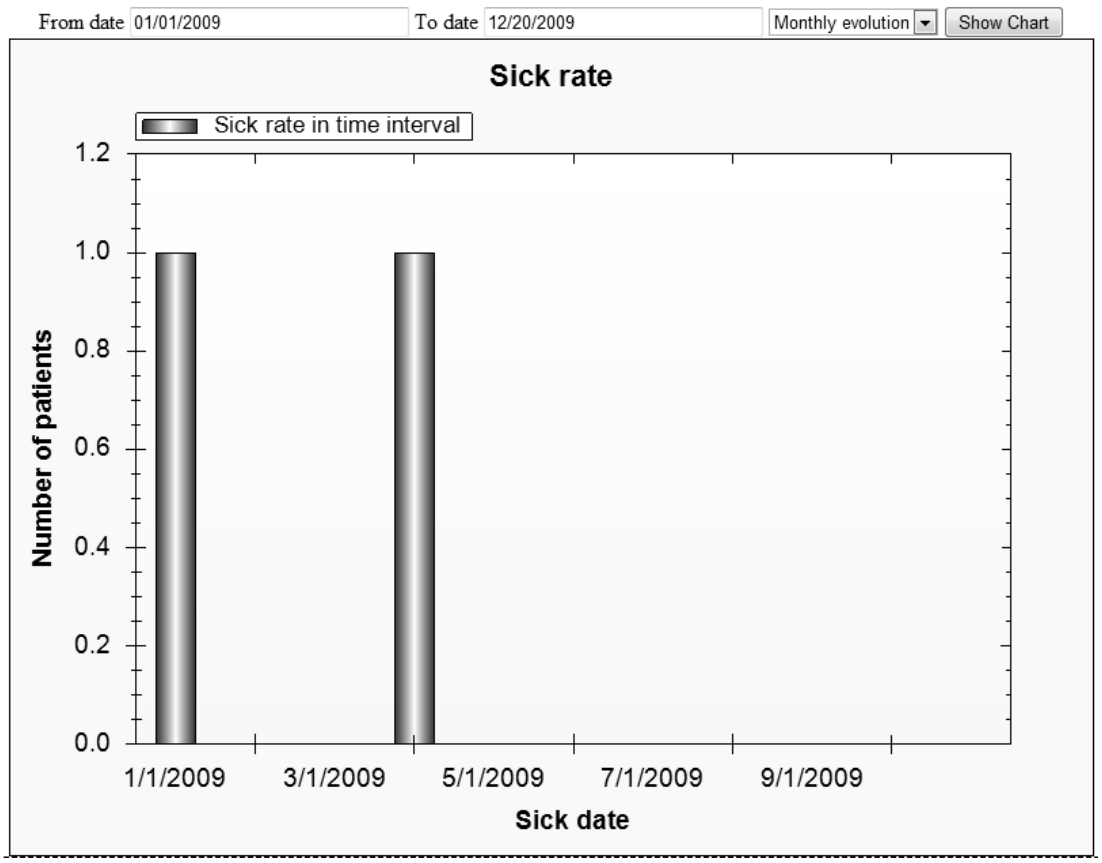

From date 01/01/2009 To date 12/20/2009 Monthly evolution Show Chart

Fig. 4 Monthly evolution for 2nd dataset

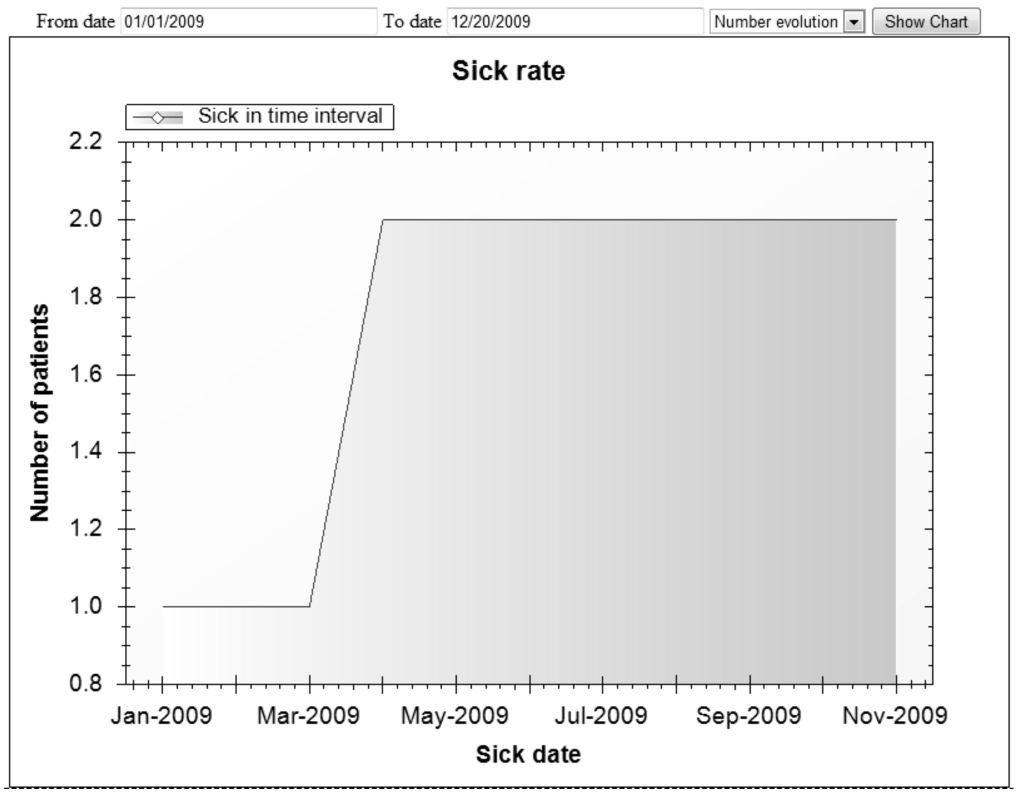


**Fig. 5 Patient number evolution for 2nd datset**

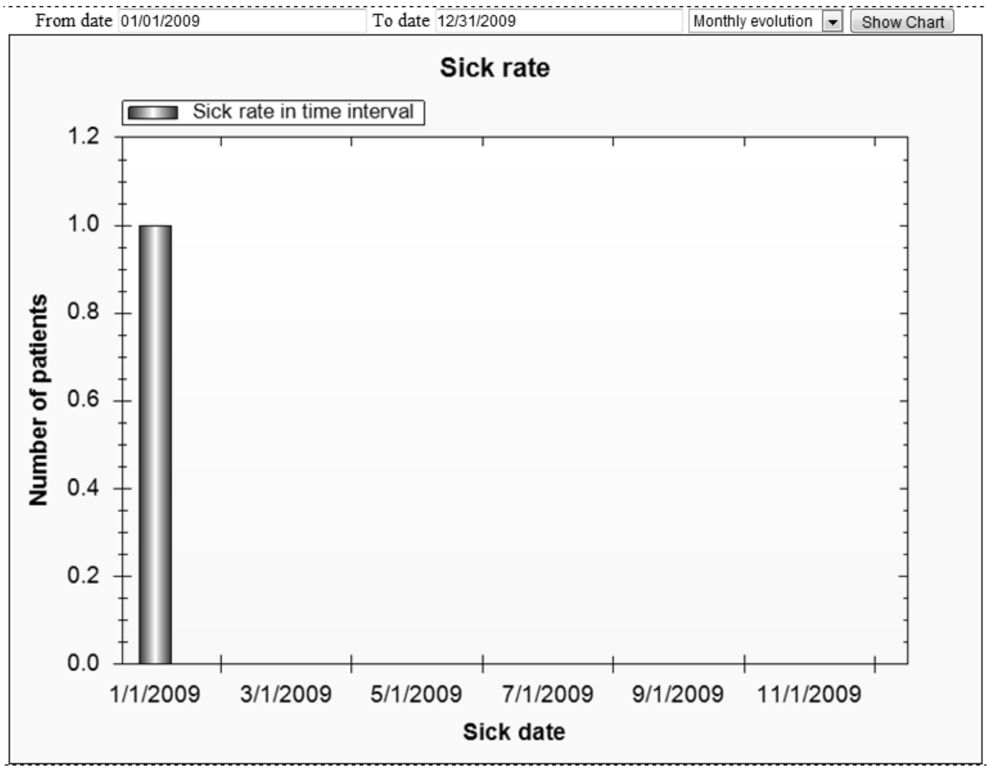


**Fig. 6 Monthly evolution for the 3rd dataset**

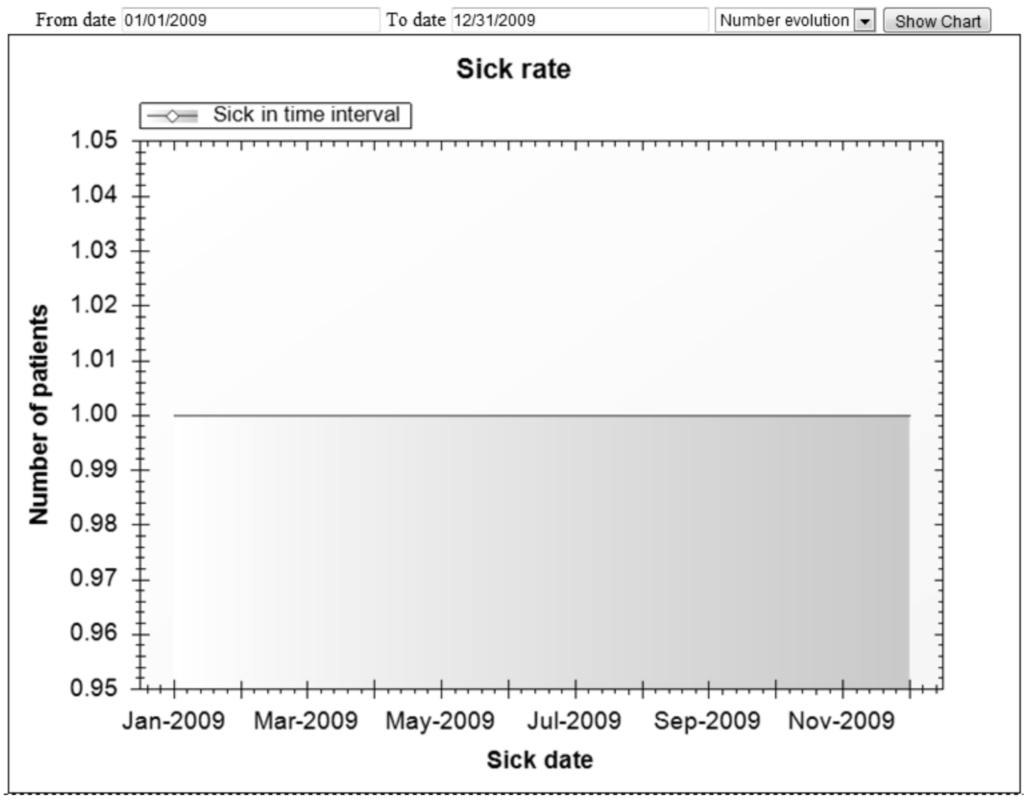


**Fig.7 Patient number evolution for the 3rd dataset**

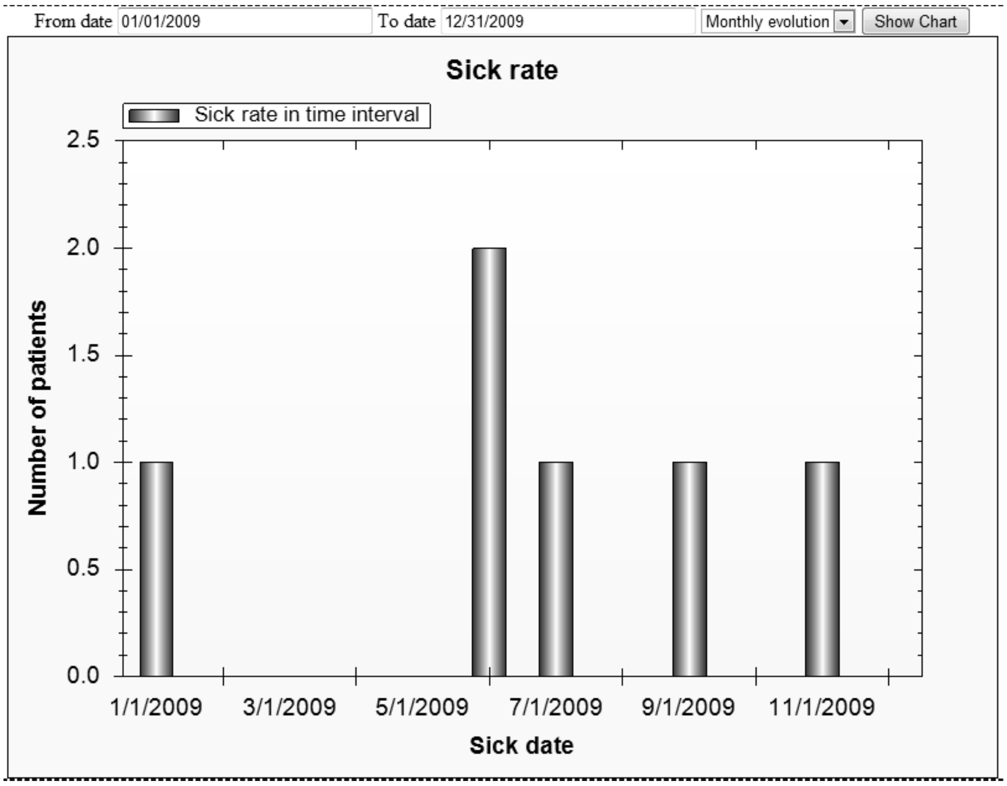


**Fig 8. Monthly evolution for 4th dataset**

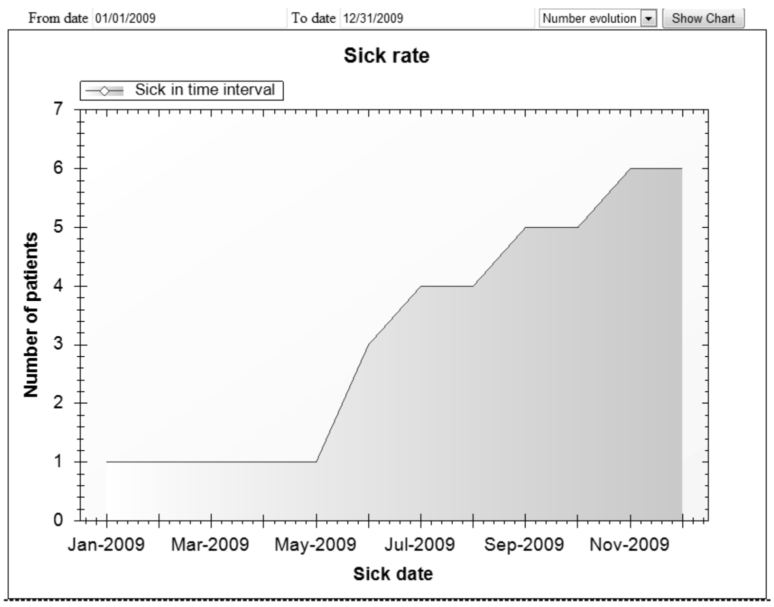


**Fig. 9 Patient number evoliution, 4 th dataset**

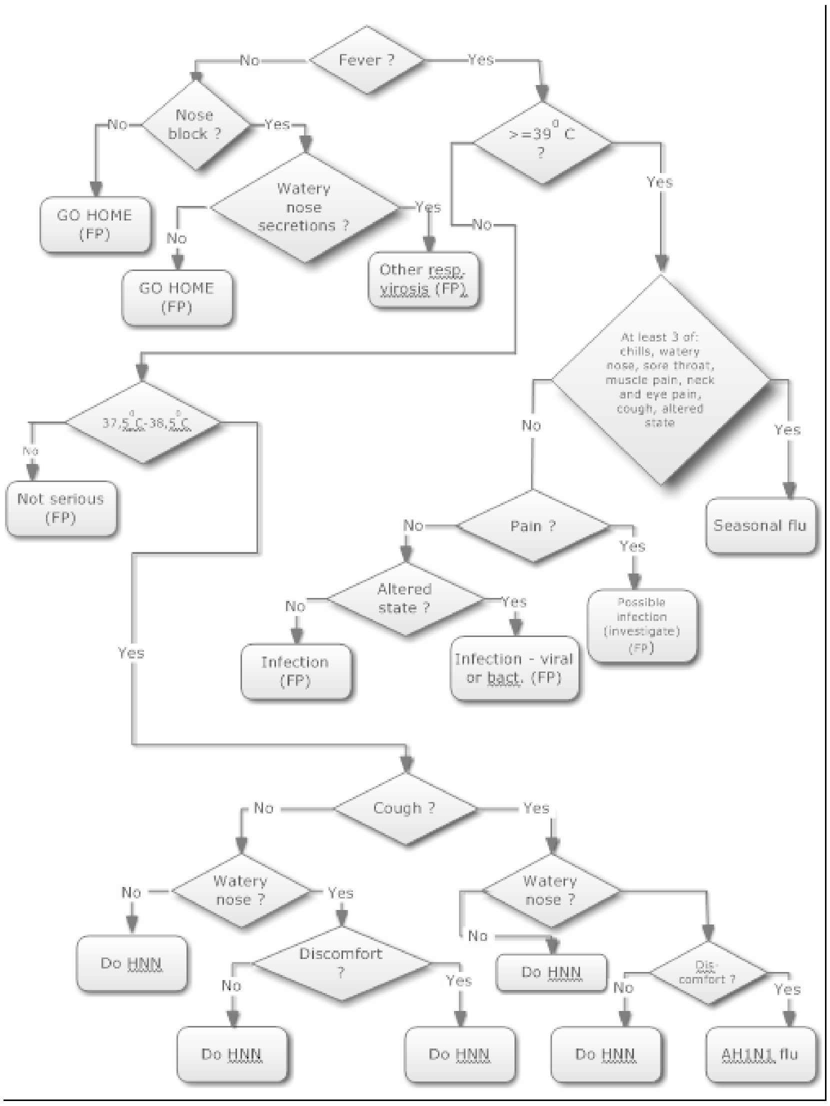


**Fig. 4 Differential diagnostics**

The first page of the application as we mentioned above is the login page, where the doctor can insert his credentials and after being authenticated (his access to resources is determined, for example he can have administrative rights, or, for the sake of the integrity of the data base, a smaller amount of control could be assigned).

The application was designed as user friendly as possible, with explanations at every moment of the execution. The next step for the doctor is to insert the patients in the database with their presumed diagnostic – it is worth noting that a diagnostic is confirmed only after thorough clinical examination and paraclinical examinations (HNN analysis, which can point accurately to the virus strain, whether it is A(H1N1) or another).

A complex logic (business logic layer) to determine, of course with the help of the doctor (who decides on the need for certain laboratory tests), whether the patient is suffering from A(H1N1) flu or the case is a false positive (any other condition). This is accomplished by continuing to the page where the application collects all the data from the patient (age, symptoms in the beginning, actual manifestations, whether it is an infection or not).

The main advantage of the approach is that by including the possibility for the doctor to correct data inserted, the integrity and reliability of the data are increased, not to mention obtaining a more natural implementation.

The main symptom which makes the doctor consider a possible infection is fever. In the case of the common flu, the fever is equal or greater than $39^0$ C, while in the case of the A(H1N1) flu the fever is between $37,5^0$ C and $38,5^0$ C. The diagnostic algorithm considers also the case of the common flu, as the symptoms are quite similar as we have already mentioned and there is a need to disseminate by further investigations the false positives from the more threatening AH1N1 infections.

In the cases when fever is at least $39^0$ C the doctor must ask the patient whether he/she has any pain, in which case it might indicate a respiratory virosis should other symptoms occur (such as chills, altered state, watery nose secretions, sore throat, etc.). When there is additional muscle or eye pain we have a case of seasonal flu.

More importantly, the case when the temperature is in the interval mentioned before, is indicating the need for a further investigation; should any of the symptoms cough, discomfort, watery nose secretions, or throat discomfort a HNN analysis is in order (AH1N1 infection is highly possible).

Returning to the main page of the application, there are a few functionalities that need attention: searching and chart generating. In order to perform the search, the instructions in the header of the page must be read; in the paper we presented charts generated by the Zed Graphical library (.dll) integrated with the application (figures 2 and 3). The chart engine needs to have as input the period on which it has to draw the charts, and the output is the monthly view of the situation and also the patient number statistics, in case the general management needs such reporting. The Zed Graphics library was chosen for its .Net platform integration oriented design.

## 4.Conclusion

As we have seen from the paper, the vast .NET platform (containing quite a lot of libraries from a broad range of development areas) can be employed even in the medical diagnostics process, in order to aid the doctors decide upon the most accurate possible verdict and decisions; the application can be considered a decision support system, as it makes the data collected about the patients available at all times for the doctor to review the decisions and eventually make corrections.

## Acknowledgement

This work was supported (in part) by the grant no. POSDRU/6/1.5/S/14b.

*References:*

[1] Sinha, Manish, Swine flu, *Journal of Infection and Public Health*, Vol.2, No.X, 2009, pp. 157—166.

[2] Buck-Emden, Rudiger, Galimow, Jurgen, *Sap R/3 system. A clien/ server technology*, Addison-Wesley, 1996

[3] Ebada Sarhan, Atif Ghalwash, Mohamed Khafagy, "*Queue Weighting Load-Balancing Technique for Database Repl. in Dynamic Content Web Sites*", Proc. of the 8th WSEAS Int. Conf. on Comput. Int., Man-Machine Sys. and Cybernetics (CIMMACS '09)

[4] Hsaio-Fan Wang, Cheng-Ting Wu,"*A Strategy-Oriented Oper.Module for Recommender Systems in E-Commerce*", Proc. of 9th WSEAS Int. Conf. on Applied Inf. and Comm. (AIC '09)

[5] Akshai Aggarwal, Sujata Kanhere, Vishnu Kanhere, Shankar Kanhere, "*The 4th Dimension of Inf. System Audit and Security",* Int. Conf. on Soft. Eng., Parallel and Distr. Systems (SEPADS '09)