# Input and Sharing of Infectious Disease Data at the Grassroots Level
# TJHSST Senior Research Project
# Computer Systems Lab 2009-2010

Anna Stapleton

May 28, 2010

## Abstract

This project aims to create a user-friendly system to input, manipulate and view patient data. The system involves a MySQL database being manipulated by a user interface coded in Java. Such a system is useful both for the maintanance of patient records within an individual clinic or hospital as well as the sharing and networking of data on a regional level, such that the data can be used for disease surveillance. The second aspect of this project is to test network capabilities and limitations using a NetLogo model.The long-term goal is to implement this system in rural regions of sub-Saharan Africa using laptops such as those used by the One Laptop Per Child initiative.

## 1 Introduction

The purpose of this project is to create a user-friendly database and interface which can be used to enter, manipulate, and view pertinent data on individual patient case reports. Electronic patient records systems are commonly implemented in the United States; however, the goal there is most often keeping records for individual patients, not disease surveillance and sharing of information. This system will allow for easy networking between individual computers and databases while maintaining patient confidentiality.

## 2 Background

The idea for this project was inspired by an interest in disease surveillance in areas with little advanced technology. Therefore, the program must be simple to implement on basic computers and easy to use, even for people with minimal experience using computers. Additionally, it is important that information be gathered in a way that is pertinent to both patient care and large-scale disease surveillance. Background research involved determining what specific fields of information were necessary to acheive these goals, as well as ways to avoid common pitfalls in electronic health records systems. One major concern is that the use of single-word diagnoses does not fully and accurately reflect the reality of individual cases. For example, seeing "Malaria" written as a diagnosis on a patient record does not give important information about severity, specific symptoms, pre-existing conditions, and alternative diagnoses and why they were rejected. Such information is crucial to keeping useful patient records, as well as for detecting widespread patterns for disease surveillance.

## 2.1 Goals and Guidelines

To ensure that this system would meet its goals, a series of principles was established to serve as guidelines for development. Some of these guidelines were inspired by those used in developing Windows 7 (Harris) while others were created specifically with this project in mind.

The first of these guidelines was that all software used in the program must be open source and common usage. This is to ensure that the system would be easily expandable, such that it does not take a lot of effort to set up the system on a new machine. This also serves to reduce costs of development and implementation, a key factor given that this system is primarily meant for use in developing areas which are not economically well-off. The fact that all the software is common usage means that any scientist with a basic background in computer science will likely be familiar with the programming necessary to manipulate the database. For individuals in the field, for whom this may be the first experience with computers, this property also means that training in the use of this system would be applicable to other computer usages.

Secondly, the program must have a small footprint and be efficient. In this case, efficiency refers to the time and effort necessary to use the program. In a medical setting, it is important that no time be wasted by any sort of unnecessary features. The goal in designing the user interface should therefore be to make it straightforward and intuitive, rather than to include clever design innovations which may be cumbersome to the user. The idea of a small footprint refers to minimizing the amount of software as well as hardware needed to implement the system. Ideally, the system would only require the installation of a single program onto a Linux computer in order to work.

A third priority is that there be consistency between the varying levels of the system. The goal is to integrate data from a local level into a larger system. In order to do so successfully, there must be consistency in design between the components of the user interface, as well as between the user interface and the formatting of the database itself. This is a key component in achieving the goal of combining a local record-keeping system with a large scale surveillance system.

Perhaps the most important principle is that this system must be user-friendly and useful to all users. It is key to remember that users at the local level will have little to no prior experience with using computers. It is therefor imperative that this system be intuitive to use and as simple to learn as possible. There must also be incentive for clinics to be willing to implement this system. While the long-term uses for an electronic disease surveillance system are clear, there must also be a short-term use at the local level. A system which does not aid or interferes with patient care will not be attractive to local clinics, and will not be implemented on a broad scale. The usefullness of this program as a disease surveillance system is therefor dependent on its usefullness as a patient records system.

## 3 Methodology

The project is based around the manipulation of a MySQL database. The main focus is the creation of the user interface, which will provide for both inputting and viewing of patient data. The user interface was developed in two distinct phases. In the first phase, an interface was designed using PHP and HTML. However, it was later determined that creating the interface in a non-webbased language would reduce the amount of software necessary to implement the system, thereby complying with the goal of maintaining a small footprint. The second-phase version of the program thereby consists of a user interface

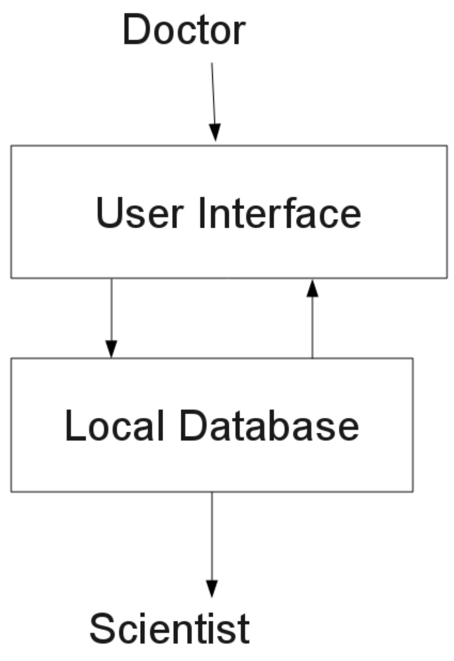

Figure 1: A basic flowchart for the communication of information

coded in Java, as well as an updated database design.

## 3.1 Phase 1: PHP and HTML Interface

The basic framework of this interface is a "report form," modeled after hard-copy patient report forms. The user fills in basic patient information (name, age, etc.) as well as clinical information (diagnosis, lab confirmation, fatality). When the user submits this information, it is entered into a single MySQL database. Other portions of the user interface allow the user to search for an individual patient record by name, and then update that record as necessary (i.e. if a diagnosis has now been confirmed by the lab). Once the basic input/search/update functions was established, additional pages were added to allow for output of information compiled from the entire database. For example, this could include allowing a user to search for the number of cases of malaria diagnosed in October 2009, and compare it to the number of diagnoses for the same month the previous year.Each of these functions consists of a single webpage or series of webpages which could be accessed from a home-page.

There were, however, several weaknesses to this design. As previously stated, it was decided that a web-based interface was not efficient in terms of software necessary to run the program. This design would require the installation of what is called a LAMP system on each computer. This system involves Linux, Apache, MySQL, and PHP. In a non-web based system, Apache would not be necessary. Instead, it would only be necessary to install MySQL and Java onto a Linux computer. This supports the goal of minimizing the amount of software necessary to implement the program.

## 3.2 Phase 2: Java Interface and Linked Databases

For phase two of the project, the user interface was switched into Java. In recreating the interface, other design flaws of the first version were also considered and addressed. The new interface presents as a series of windows, each asking for a specific category of information. Checkboxes were introduced for response to yes/no questions, such as whether the patient is experiencing a specific symptom.

Perhaps the most important improvement in this version was the redesign of the database itself. Instead of a single table containing all of the patient information and diagnosis information, the database now consists of two distinct tables which are linked. In the first table, data is stored about the patient; for example, name, age, gender, and home address. Each patient is also automatically assigned an identification number specific to that individual. In the second table, information is stored on a visit-by-visit basis. Each time a patient comes in, a new entry is created in the second database with that patient's identification number. The entry includes diagnostic information, such as the symptoms being experienced, the date of symptom onset, and

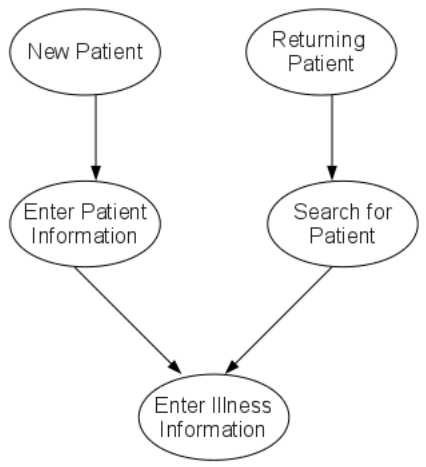

Figure 2: Input of data in the phase 2 user interface

any lab work that was done on the patient. In this way, medical workers are saved from having to redundantly enter the same basic information every time the same patient returns. This system also enhances patient care by making it possible to review a patients history with ease. Therefore, this redesign of the database fits with the aim of making the program useful at all levels of implementation.

### 3.3 Phase 3: Modeling the Network

The final step in the project was to examine how the system might work in the field. At the core of this project is the ability to network information both between individual clinics and from clinics to scientists and epidemiologists. The system is to be implemented using the XO computers created and sold by the One Laptop Per Child Initiative, which have the ability to mesh network. Due to funding limitations, it is not possible to obtain a pair of these computers in order to test their networking capabilities and compatibility with this program. In place of that test, a model of mesh networking was conducted in NetLogo, in order to optimize implementation for minimal costs while maintaining the effectiveness of the system. This information can later be used in the geographic design and distribution of the system, in order to maximize the abilities of the system while minimizing costs.

The model was designed to demonstrate the effects of levels of connectivity. The network consists of a group of nodes, each of which represents a laptop possesing a local-level database. One node was designated the "master node," and represents the central computer collecting data from the various local nodes. In the model, the nodes maintain a constant location, while the master node travels with each "tick," or iteration of the process. Nodes are able to connect automatically when they come within a certain distance of each other.

Each node also possesses a group of variables which describe its ability to successfully transfer information to the master node. First, the node knows whether or not it is connected to the master node. Nodes can be connected either directly or indirectly. A directly connected node is said to be one "hop" from the master node, meaning there is a single link connecting the node to the master (i.e. A to M). An indirectly connected node is one which is multiple hops from the master, meaning it is connected to another node which in turn is connected to the master (i.e. A to B to M). Secondly, all nodes also know their level of connection: a node which is in no way connected to the master node is level 0, directly connected nodes are level 1, nodes two hops away (i.e. A to B to M) are level 2, nodes three hops away (i.e. A to B to C to M) are level 3, etc. Finally, each node stores its physical distance from the master node on the 2-D plane.

These variables are then used to calculate the likelihood of a node successfully passing its information to the master for each tick. According to the literature, two of the main criteria which limit the capabilities of the network are the physical distance between nodes and the levels of connection between nodes. Rastogi et. al. found that nodes could connect reliably within 150 meters of each other, and provided a graph of the variation of successful throughput with distance. From this graph it was possible to derive a quadratic equation to calculate the likelihood that a node could pass its information based on the total distance between it and the master node. Rastogi et. al. also described that the transfer rate for a two-hop (level two) connection was 70 percent that of a direct connection. This information was extrapolated to calculate the decreasing chance of a successful pass with an exponential equation. Each node possesses a variable called "pass-chance." If the node is completely disconnected from the master node, its pass-chance is set to 0. If the node is connected directly or indirectly to the master, pass chance is calculated as follows:

$pass - chance = (.837^l) * (.00008 * ((d/l) * 30)^2) - .021 * (((distance)/l) * 30) + 1.417)$

Where $l$ is the level and $d$ is the distance between the node and the master node. Distances are multiplied by a factor of 30 because of the scale of the model, where one unit distance is equal to 30 meters. Based on this formula, each node will have a pass-chance value between 0 and 1. If the pass-chance value for a node is greater than .5 (representing a 50 percent chance of success), the node is considered to have successfully passed its data to the master node. Once a node has exceeded a pass-chance of .5 once, it maintains its status of having successfully passed data for the remainder of the 365-tick trial period. This represents the goal of trying to collect information from every node at least once yearly.

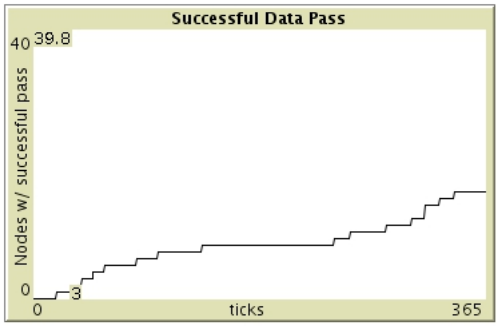

Figure 3: A sample ouput of data; number of nodes with successful information pass relative to the number of ticks

## 3.4 Testing the Interface

The testing of the user interface is based on checking for compatability with the goals layed out in the beginning of this paper. In this sense, testing was conducted following the first phase of developing the user interface. The first design failed the test of these goals on multiple levels. First of all, the design did not minimize the amount of software necessary to implement the program. Secondly, the design of the database required redundant data entry, making it too slow and clumsy to be useful to clinic workers and patients. Finally, certain aspects of the layout of the electronic entry form, such as that all the information was requested on one long page, made the interface intimidating for users who have little or no experience with computers. In the second phase of the user interface design, these problems were addressed in multiple ways. The system was switched from PHP and HTML into Java, reducing the amount of software needed to run the progam. In the Java version, the interface was broken into multiple smaller screens, reducing the amount of information requested at any given time. The database was then recreated in order to remove the redundancies of entering basic patient data for multiple visits of the same patient.

## 3.5 Testing the Model

Testing of the networking function using the NetLogo model yields more quantitative results. The main output of the program is the number of nodes which have successfully passed information to the master node relative to the number of iterations of the algorithm. The goal of testing was to search for a correlation between the density of nodes and the successful pass rate. For each experiment, node density (ND) was calculated to be the number of nodes divided by the world area, where world area equals world width times world height. The successful pass rate (SPR) was calculated as the number of nodes to have passed their information divided by the total number of nodes.

Three sets of testing conditions were run. In the first group, the World Area was held constant at 1024 square units (corresponding to 921,600 square meters), while the number of nodes was varied. The number of nodes was set to 50, 40, 30, and 20 in succession, with 10 trials being run for each set of conditions. For each trial, 365 ticks were run, and the program outputted the number of nodes which had passed their information. From the data, the average successful pass rate for each condition was calculated and plotted against node density (see Figure 4).

Data from the first round of trials made clear that a system of those dimensions would have very low success rates. The highest node density tested in that scenario was .031, corresponding to about one node per 32 square meters. The average SPR for that density was .078, meaning that about one in twelve nodes successfully transfered data. In the field, such a remarkably low success rate would be unacceptable and ineffective in providing useful data on infectious disease activity.

Because of the inadequacy of the first set of conditions tested, the second set of tests were designed to show the high density of nodes necessary to achieve an acceptable SPR. The World Area was again held constant at 1024 square units, while the number of nodes was set to 75 and 100, with the same number of trials per condition set being used as in the first round of tests. With a node density of .098 (corresponding to the 100 nodes condition), the average SPR was .475. When the node density was set to .073 (with the 75 nodes condition), average SPR was .571. This suggests that in order to have one of every two nodes successfully pass their information, the system would need to have one node per every 10 square meters. Not only is this still an unacceptably low success rate, but the density of nodes involved renders the system unrealistic.

The final round of testing was designed to demonstrate the success this system might achieve given a more realistic set of constraints. The World Area was set to 9801 square units (the largest allowed in NetLogo), corresponding to 8,820,900 square meters. For reference purposes, this is roughly a third of the land area of Rwanda. Supposing that there were to be one node per every 100 square kilometer area, the number of nodes was set to 88, meaning a node density of about .009. Under this set of conditions, the average SPR attained was .038, meaning that approximately one in every 27 nodes successfully transfered its data.

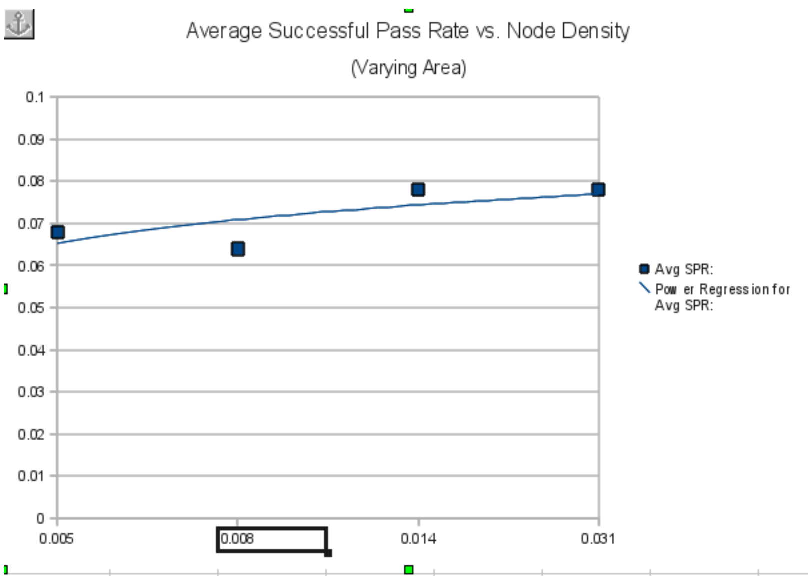

Figure 4: A plot of the average SPR per node density from the first round of trials

## 4 Expected Results and Value to Others

### 4.1 Immediate Results

The combination of the easy-to-use Java interface with the linked database system allows for a simple data-entry process. The program complies with design goals for a useful, straightforward system. Use of Java to generate the interface makes it possible to implement the system on virtually any machine with minimal installation requirements. The use of linked tables in the database prevents data entry from encumbering medicals workers and reducing the quality of patient care. In fact, the use of this system in a rural clinic may actually help to improve patient care at the local level, because many clinics currently have little or no capability for the keeping of consistent patient records. Finally, the simple design of the user interface makes use of this system intuitive for individuals who have had little or no previous contact with a computer.

## 4.2 Ideas for Improving the Model

Several aspects of the NetLogo model could be improved in order to produce a more accurate representation of the system. One improvement would be to introduce stochasticity at different levels of the calculations. For example, as the model stands, any node with a pass chance of greater than .5 will be considered successful in passing its information. In reality, a fifty percent chance of success obviously does not guarantee sucess, nor does a less than fifty percent chance guarantee failure. Allowing for stochasticity at that level would create a more realistic scenario when determining whether a node can successfully pass its information.
A second critique of the current model is that it does not currently take into account the amount of time a node stays in contact with the master node. For example, node A may be connected to the master node such that it has a pass success rate of .45. Because this is less than .5, node A would not successfully pass its information, even if it maintains a .45 pass success rate for multiple ticks. In reality, the longer a node is in contact with the master node, the more opportunities the node has to pass its information, and there is therefore a greater chance of one of those attempts succeeding. A more accurate model of this network would need to be able to track the amount of time a node has been in contact with the master, and readjust its pass success chance accordingly.
Thirdly, the current model does not represent a mesh network's ability to propagate shared information. For example, suppose node A is linked directly to nodes B and C. Node B would automatically pass all of its information to node A, and node C would do the same. At this point, node A possesses all the information for nodes A, B, and C. This means that if node A can successfully transfer information to the master node, it will transfer not only its own information, but the information of nodes B and C, even if nodes B and C are not within range to pass the information themselves. In order for this to be properly reflected in the model, each node would need to calculate a pass chance for every node it connects to, not just the master node. The fact that this factor is not incorporated into the model means that the data produced is likely to be a gross underestimation of the actual average SPR.

## 4.3 Long-Term Goals and Uses

This project is meant to be implemented in rural clinics in sub-Saharan Africa, where there are high incidence levels of multiple infectious diseases. In the long term, it is hoped that the simple computers used by the One Laptop Per Child initiative, which are ideal due to their low cost and capability for mesh networking, can be distributed to clinics and used by doctors to communicate information about cases of infectious diseases both amongst each other and with the scientific research community. The program doubles as both a disease-surveillance tool and a basic electronic patient records system. Therefore, implementation of the system would lead to higher quality patient care on two levels. At the local level, easy access to patient records allows medical workers to consider a patient's history when making a diagnosis. On a broader scale, availability of consistent, current data on disease incidence levels will allow scientists to better predict, prepare for, and respond to major outbreaks of infectious disease.

## 5 Acknowledgements

Thanks to Dr. Michael Johansson for inspiring and guiding the development of this project, and for being a great big brother.

## 6 Bibliography

Buchele, S. F. "Using OLPC Laptop Technology as a Computer Science Case Study Teaching Tool." Consortium for Computer Sciences in Colleges. 2009.

Harris, Jensen. "MIX08 Microsoft Office 2007: The Story of the Ribbon." Posted 11 January 2009. ¡http://www.youtube.com/watch?v=Tl9kD693ie4¿.

Johansen, M. A. Scholl, J. Hasvold, P. Ellingson, G. and Bellika, J. ""Garbage in, garbage out": extracting disease surveillance data from epr systems in primary care."

Kraemer, K.L, Dedrick, J., and Sharma, P. "One Laptop Per Child: Vision vs. Reality." Communications of the ACM, Volume 52, No. 6. June 2009.

Rao,R. Krishnan, S. Niculescu, R. S."Data Mining for Improved Cardiac Care." SIGKDD Explorations, Volume 8, Issue 1. 06/11/2006

Rastogi, V., Ribeiro, V. J., and Nayar, A. D. "Measurements in OLPC Mesh Networks." Proceedings of the 7th international conference on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, 2009. Seoul, Korea.