Modeling and Implementing Record Linkage in Health Information Systems

Philip Achimugu, Babajide Afolabi, Abimbola Soriyan

© *The Journal on Information Technology in Healthcare* 2009; **7**(3): 169–181

Modeling and Implementing Record Linkage in Health Information Systems

Achimugu Philip, Soriyan Abimbola, Afolabi Babajide**

HIS R & D Unit, Computing Centre, *Department of Computer Science & Engineering, Obafemi Awolowo University, Ile-Ife Nigeria.

ABSTRACT

Background: The process of identifying record pairs that represent the same entity (duplicate records) is technically known as record linkage and is one of the essential elements of data cleansing. This paper proposes a fast and efficient method for linkage detection within the healthcare domain. The features of the proposed approach are an embedded fast blocking method with a string matching function that accounts for keystroke mistakes made during data entry of a patient's name and the addition of a module that dynamically generates blocks of possible associated and unique records.

Methods: The proposed methods for achieving our goal are as follows. The first step is to standardise the data in the database using SQL (Structured Query Language) statements. The second is to match similar pair records, relying on the discretion of a human expert and observation inferred from the decision support system. The final step is to organise records into match and non-match status. The system was developed in Unified Modeling Language and was coded in Java Programming Language.

Results: It was observed that the linkage algorithm with embedded string matching function catering for subtle errors in strings, performs better in duplicate detection (40% recall and 60% precision) than linkage algorithm without a string matching function (90% recall and 10% precision).

Conclusion: In this paper we have described an improved record linkage system based on an embedded fast blocking method with a string matching function. The system can be used to improve the data quality and integrity of electronic healthcare records.

INTRODUCTION

Many private and public healthcare organisations capture, store, process and analyse a rapidly growing amount of patient data contained within millions of records. This data consists of both demographic and clinical information. Linking and aggregating records relating to the same person from several databases is becoming increasingly important. Information from multiple sources can be aggregated to co-ordinate care,

Correspondence and reprint requests: Achimugu Philip, HIS R & D Unit, Computing Centre, Obafemi Awolowo University, Ile-Ife. E-Mail: check4philo@yahoo.com.

giving treating clinicians a comprehensive overview of a patient's health status and treatment. This can improve the quality and safety of patient care and the accuracy of data used for reporting. The aim of such linkages is to match all records relating to the same entity.

The basic method compares name and address information across pairs of files to determine those pair of records that are associated with the same entities. More sophisticated methods use information from multiple lists[1] to create new functional relationships between variables in two files that can be associated with new metrics for identifying corresponding entities[2] or use graph theoretic ideas for representing linkage relationships as conditional random fields that can be partitioned into clusters representing individual entities[3]. Whichever technique is chosen, the principle of a record linkage process is to try to pair similar records to avoid duplicates. This is important as duplicate records can results in treating clinicians missing vital patient information which is in one record and not another. It can also have the effect of erroneously inflating estimates of the number of entities in different categories[1]. For example if a rare disease is documented and coded in 2 separate records for a patient, then a report pulled for all patients with the disease will count that person twice rather than just once.

The main challenge in this task is designing a function that can resolve when a pair of records refers to the same entity in spite of various data inconsistencies. Data quality has many dimensions one of which is accuracy. Accuracy is usually compromised by accidental errors introduced in a database system. These errors result in inconsistent, incomplete or erroneous data elements. In order to improve the accuracy of the data stored in a database system, it is necessary to be able to compare it with other data stored in the same or a different system.

With health related data spread between or across many administrative and clinical databases, the ability to bring the data together dynamically is important. At present patients are seen in different health organisations who own their own records and even in the same healthcare institution a patient may have 2 sets of records. For example a different record might be created by 2 different departments or a new record created for different admissions. In addition multiple records for an individual patient may be created as a result of spelling mistakes, e.g. Smit and Smith or differences in registration of names e.g. William Smith, Will Smith, Bill Smith or changes in names, e.g. a female getting married and taking her husband's surname. A single health record is essential for optimal clinical decision making, and is also vital for supporting the accuracy and validity of administrative reporting, clinical research and data mining. Record linkage is therefore important in healthcare and its efficient implementation can enhance an organisation's performance.

The objective of this paper is to develop a record linkage system that is capable of enhancing the detection of duplicate records in a health information system.

METHODS

The overall schema of the process for detecting duplicate records is shown in Figure 1.

The database used for the study was MINPHIS (Made in Nigeria Primary Healthcare Information Systems). This was collaboratively developed by the Health Information Systems Research and Development Unit of the Computing Centre of the Obafemi Awolowo University, Ile-Ife and the Health Information Systems Research and Development Unit of the University of Kuopio, Finland and is currently deployed to over 15 teaching hospitals in Nigeria.

The first step in the process is to access the patient records in individual MINPHIS databases (Figure 1) and subject them to a cleaning and standardisation process. This was achieved using Java Database Connectivity (JDBC). JDBC is an Application Programming Interface (API) for the Java programming language that defines how a client may access a database. It provides methods for querying and updating data in a database. It is oriented towards relational databases and allows multiple implementations to exist and be used by the same application. The test used a simplified dataset containing all health data information of individual patients. For the purpose of this research study the attributes extracted from the database for experimental evaluation were Surname, First Name, Sex, Date of Birth and Address. Records representing the same entity based on the static decision rules were given the value of 1 while those that are not were given the value 0. This was the criteria for duplicate detection.

The data extracted from the MINPHIS database is then subjected to a standardisation process so that the total number of records in the database is calculated

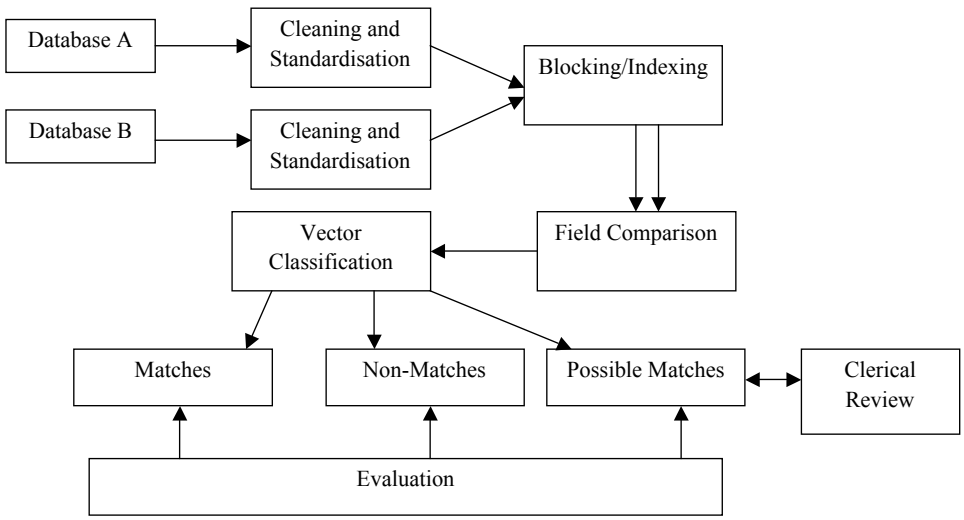

Figure 1. *Overall schema Classification of Pairs of Records*

and displayed in a convenient format. This is achieved through the SQL (Structured Query Language) command which is executable in the cache database that MINPHIS uses (the cache database has embedded SQL statements in it). The data can then be subjected to the "find duplicate" process (Figure 1).

To find duplicate records, the records are initially automatically "blocked". A block is a collection of records that potentially duplicate each other. The records are displayed in alphabetical order and only records that fall within the same block are compared to determine their match or non-match status. When a single record is clicked on, that record undergoes a string matching process that compares all other records that fall within that block before it displays likely or suspected duplicates. Although this process is automated the final decision as to whether a record is a duplicate is made by a human specialist. Two records are said to be duplicates if they are semantically equivalent; if they are not, the data is migrated to a non-duplicate table. A use case for the system is shown in the Appendix.

The technologies underlying the system will now be described in greater detail.

Output Files

Before a linkage or reduplication project can be started, the ways the match status and the non-match status will be saved into files have to be defined on the output/run page. There are also two further output options that can be set. The first is the percentage value according to which a progress report for the various steps in the linkage process will be reported. For example, for each percentage of record pairs compared, a progress message will be given. The second option allows for length filtering in the field comparison step. This allows efficient filtering of record pairs. In practice all non-duplicate record are stored in the non-duplicate table and the records that are duplicated are skipped by the system. In this way only new data is migrated to the new table.

String Matching Function

Dealing with typographical errors is extremely important in a record linkage context. If comparisons of pairs of strings are only done in an exact character-by-character manner, then many matches may be lost. Previous studies have shown that almost 20 percent of last names and 25 percent of first names disagreed character-by-character. If matching had been performed on a character-by-character basis, then more than 30 percent of matches would have been missed by computer algorithms that were intended to delineate matches automatically. In such a situation, manual review and (possibly) matching error would have greatly increased[4,5].

In our project we have introduced a string matching function in names of candidate pairs that accounts for keystroke mistakes during data entry. The basic concept is to compute the string lengths, find the number of common characters in the two strings, and find the number of transpositions.

Algorithms

Two algorithms were utilised in this work. The first is the Sorted Neighbourhood Method Algorithm (SNM) and the second is the Modified Algorithm which has embedded String Matching. This incorporates a string matching function that caters for typographical errors in candidates' names which may occur as a result of data entry errors or unclear spellings of names.

The basic premise of the SNM approach is that records are lexicographically sorted using some attributes and potential duplicate records are located in neighbourhoods.

During the record linkage process, because the potential number of records to be compared is large, record linkage systems employ a technique called blocking, i.e. grouping records into collections of records that potentially duplicate each other. This technique is used in order to reduce the total number of records to be compared so that only records within the same block are compared. The decision as to what makes two different records duplicates can be determined in a number of ways. A basic way is to compare the key fields of a sample record to its counterpart in the reference record. If a binary equality is obtained, it can be assumed that the entities are the same. To increase efficiency, the comparison rule can be enhanced. While a binary operation returns a Boolean value (1 if the fields are strictly the same and 0 if they are different), a similarity operator will return a score ranging from 0 to 1, showing field proximity and quantifying their difference. The higher the result, the nearer the field values are. Such a similarity operator is based on approximate string matching function.

In order to enhance the comparison principle, we run the approximate matching function embedded in the Sorted Neighbourhood Algorithm on different matching variable and create atomic similarities (fields of records) for the comparison of the date of birth, the first names and the surnames (birth name), if necessary between the married name (otherwise, the atomic similarity is considered as missing), if necessary between the married name of the first record and the birth name of the second (otherwise missing), if necessary between the surname of the first record and the married name of the second (otherwise missing) and one for the address information.

The basic principle of the string matching function is to compare two strings C_1 and C_2 to compute a similarity rate. Letters in the first string are considered and compared to the letters in the second string. If the letters in the first string are more than three-quarters the length of the shorter or second string, this indicates a typographical error or subtle spelling change in the second string and the algorithm returns a value 1 for similar strings and 0 for different strings after processing or comparing the rest of the fields of one record to its counterpart in the reference record.

For L_1 and L_2; C_1 and C_2's respective lengths, the similarity rate is obtained by:

$$\frac{\frac{N_c}{L_1} + \frac{N_c}{L_2} + \frac{N_t}{2N_c}}{3}$$

Where N_c is the number of common characters in the two strings and N_t is the number of transpositions. In order to evaluate the performance of the system, normalised measures, i.e. precision and recall were used.

The Sorted Neighbourhood Method (SNM) Algorithm was used to implement the record linkage systems. The SNM scheme first sorts all the entities using the preselected key attributes (Surname, First name, Gender, Date of Birth and Address) to detect duplicates. The string matching function was embedded in the algorithm to ensure that typographical errors in strings used as attributes such as names (surname and first name) of candidate pairs during the blocking process is catered for. A fixed-sized window with a slide functionality covering the beginning to the end of the list is created (Figure 2). In each sliding window, the first entity is compared to the rest of the entities within the same window to identify pairs with a small distance. Thereafter, the window slides down by one entity and the process repeats until it reaches the end of the list. Thus, the sliding window works as a blocking scheme with the premise that matching entities are lexicographically similar to those that are located within the same window.

Figure 1 illustrates the SNM. A single pass of SNM over n records with w records per window yields n – w + 1 blocks. Since each block incurs w – 1 number of record comparisons, overall, the running time of SNM becomes $0(n^2)$. Due to the relatively faster running time compared to the baseline approach and the simple implementa-

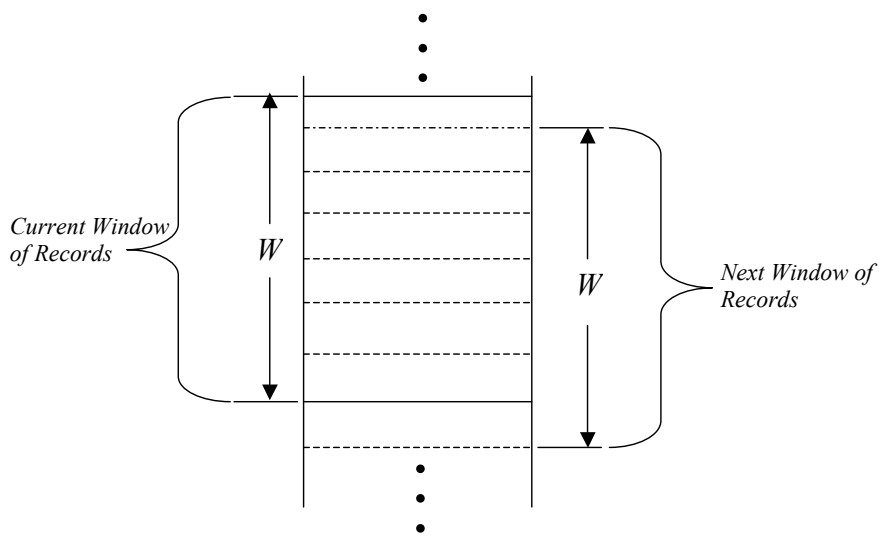

Figure 2. *Illustration of the Sorted Neighbourhood Method Algorithm*

tion, the SNM approach has been a popular choice of record linkage algorithm in many data applications.

It is of note that this simple algorithm uses several user determined parameters:

a. Choice of attributes for a key
b. Size of the sliding window
c. Distance threshold

In the algorithm, all the above parameters are pre-set once and never.

The output from the blocking step are candidate record pairs, while the comparison step produces weight vectors with numerical similarity weight. These are then classified into matches, non-matches and possible matches (Figure 1).

Evaluation of the Algorithm

The Sorted Neighbourhood Method algorithm was evaluated to determine its accuracy as a function of the parameter k (threshold of similarity) and to obtain a value for k. The evaluation of the Sorted Neighbourhood Method algorithm encompasses two main issues: the accuracy and the behaviour of the similarity threshold k. First, the distances between all possible pairs of records (ri, rj) are computed and stored in a matrix. Then, for each value of k, the total number of true positives (TP), false positives (FP), false negatives (FN) and true negatives (TN) matches are computed as shown in Table 1.

To evaluate the accuracy independently from the total number of records in the database, normalised measures known as **Recall** and **Precision** were used. Recall is the percentage of pairs representing the same entity that were detected as such by the algorithm. Precision is the percentage of pairs detected by the algorithm as representing the same entity that really represent the same entity:

$$\textbf{Recall} = \frac{TP}{TP + FN}$$

$$\textbf{Precision} = \frac{TP}{TP + FP}$$

Table 1. *Classification of Pairs of Records*

	Below the threshold "k"	Above the threshold "k"
Same "similarity code"	True Positive (TP)	False Negative (N)
Different "similarity code"	False Positive (TP)	True Negative (N)

RESULTS

The results of the evaluation are presented in Figure 2. As expected there is a trade-off between Precision and Recall when the threshold k is varied. Its optimum value is considered to be at the intersection of the two curves.

In the case of Sorted Neighbourhood Algorithm, for example, it is approximately 0.18, corresponding to 0.95 for Precision and Recall (Figure 3). In comparison with exact record matching, which is equivalent for the case when k = 0, the approximate record matching (with higher values for k) provided a good gain in Recall, without significant loss in Precision. However when the dataset of MINPHIS was tested on the algorithm with the embedded string matching function catering for typographical errors in candidates names, exact record matching obtained a Recall of only 40%, while the algorithm without a string matching function with k=0.1 obtained Recall

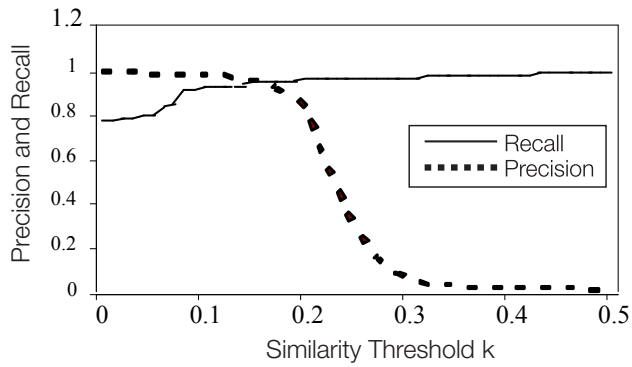

Figure 3. *Sorted Neighbourhood Algorithm*

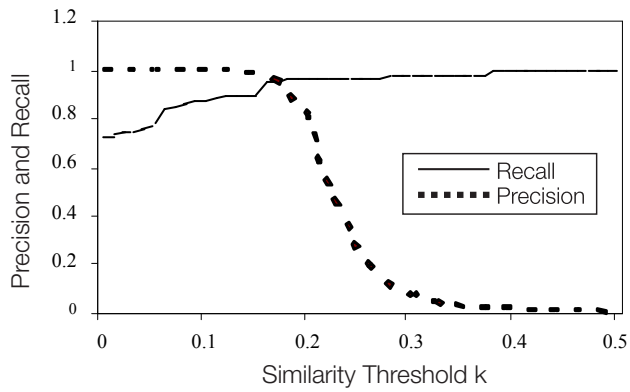

Figure 4. *Sorted Neighbourhood Method with String Matching Function*

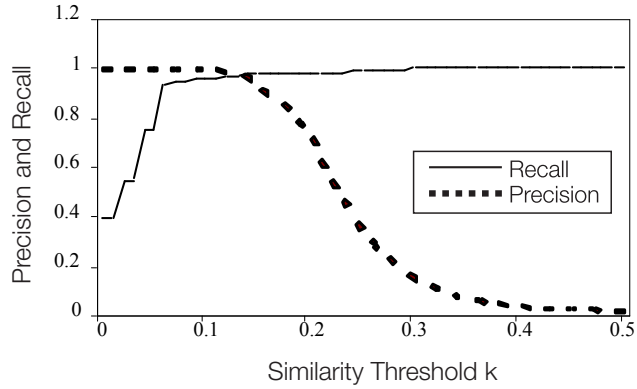

Figure 5. *Tradeoff between Precision and Recall*

greater than 90% (Figure 4). For Precision the figure was 60% for the algorithm with the embedded string matching function and 40% without.

There are also large regions $0.14 < k < 0.19$, for the first algorithm (without string matching function); $0.07 < k < 0.15$, for the second algorithm (with a string matching function); and $0.09 < k < 0.19$, for the tradeoff between precision and recall), where both precision and recall are high (greater than 0.9) (Figure 5). This allows some freedom and safety in the choice of k. However, the appropriate setting of the parameter "k" is a prerequisite for the Sorted Neighbourhood Method success. Thus, it is important for creating a process for estimating the optimal value for threshold "k".

DISCUSSION

The initial idea for record linkage was conceived by Halbert Dunn in 1946 who when chief of the U.S. National Office of Vital Statistics used the term to refer to linking vital records, such as birth and death certificates, pertaining to a single individual[6]. Computerised record linkage was proposed a decade later when Howard Newcombe and colleagues used computers to link vital records in an effort to track hereditary diseases[7]. The theory of record linkage was further expanded by Ivan Fellegi and Alan Sunter who demonstrated that probabilistic decision rules were optimal when the comparison attributes are conditionally independent[8].

Record linkage can be regarded as having three main phases or elements:

(i) Bringing pairs of records together for comparison. In this step, it is important to minimise failure to bring together potentially linkable records while at the same time minimising the number of comparisons between un-linkable records. It is virtually impossible to carry out probability matching on all pairs of records involved in a linkage. Usually only a subset are compared; those which share a minimum level of identifying information. This has been

traditionally achieved by sorting the files into "blocks" or "pockets" within which paired comparisons are carried out.

(ii) Calculating probability weights. The second step in record linkage is a detailed comparison used to determine if the pair of records can be linked. To optimise this step there are two types of errors, accepting false linkages and rejecting true linkages, that should be limited according to the use of the program[9]. To help optimise the searching process the creation of subdivisions to arrange the files in some orderly sequence so that comparisons are only conducted on a small portion of the two files has been proposed[10].

(iii) Making the linkage decision. The final challenge is converting the probability weights representing relative odds into absolute odds which will support the linkage decision.

Of these three areas, it is the second, the calculation of probability weights which has received the most attention and is the best understood. Various methods of improving the accuracy of linkage have been described, e.g. clustering large and high dimensional datasets[11]. To achieve good linkage it has been proposed that the choice of blocking key is important and that it is preferable to use the least error-prone attributes available[12]. The method presented by Fellegi and Sunter[8] also requires careful use of frequencies from the data without the training set. They demonstrated that the weights can be computed directly from the data. However, they also state that closed form solutions from their method can only be obtained if there are three fields being compared and a conditional independence assumption is made. With more variables, the Expectation-Maximization, or EM, algorithm can be applied to obtain estimates of the parameters needed in weight computation[13].

A concern has been to avoid over elaboration and over complexity in the algorithms which calculate the weights and because the human mind is difficult to replicate, our enhanced algorithm displays the atomic information of suspected duplicates and the final decision is left for a human expert to decide. We believe that this is more reliable as human judgment is usually superior to an expert system's judgment. Beyond a certain level of increasing refinement of the weight calculation routines tends to involve diminishing returns. This relatively basic approach has been facilitated by the relative richness of the identifying information available on most health related records.

Our method gives those specialists responsible for merging similar records a representative view to show them how close records in some homogenous or heterogeneous sets are. Additionally, the algorithm and underlying database support real-time detection of duplicate records. This can help to avoid the creation of duplicate records by alerting the user that several neighbour records already exists (see Use case in Appendix). This real-time use could also be used in multi criteria searches for identities and a simple as well as easy to use front end algorithm was employed in the implementation of the record linkage system so that short response times are achieved. Response time is closely related to optimisation of the algorithm and

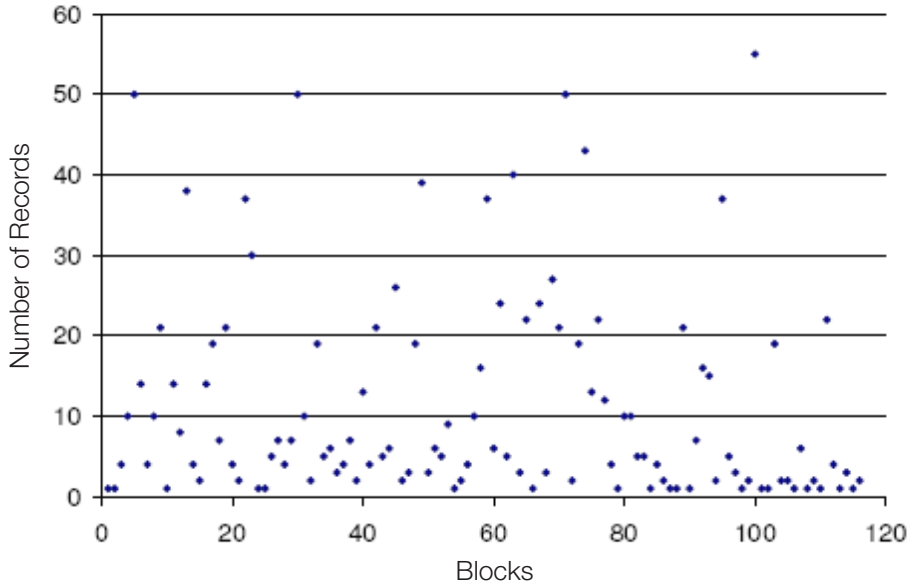

Figure 6. *Ideal block sizes in the MINPHIS dataset (116 blocks, alphabetically ordered)*

especially the blocking part. Its improvement allows the reduction in the number of potential duplicates to be tested by the main algorithm (Figure 6).

In summary in this paper we have described an improved record linkage system for health information systems which can be used in real-time.

REFERENCES

1 Winkler WE. Record linkage software and methods for administrative lists. Eurostat, *Proceedings of the Exchange of Technology and Know-How, 1999.* http://www.census.gov/srd/www/byyear.html.

2 Scheuren F, Winkler WE. Regression analysis of data files that are computer matched, II," *Survey Methodology* 1997; **23**: 157–65. http://www.fcsm.gov/working-papers/scheuren_part2.pdf.

3 McCallum A, Nigam K, Unger LH. Efficient clustering of high-dimensional data sets with application to reference matching. *Sixth ACM Conference of Knowledge Discovery and Data Mining*, 169–78, 2000. http://www.kamalnigam.com/papers/canopy-kdd00.pdf.

4 Winkler WE, Thibaudeau Y. An Application of the Fellegi-Sunter Model of record linkage to the 1990 U.S. Census," U.S. Bureau of the Census, Statistical Research Division Technical Report (1991). http://www.census.gov/srd/www/byyear.html.

5 Jaro MA. Advances in record-linkage methodology as applied to matching the 1985 Census of Tampa, Florida. *Journal of the American Statistical Association* 1989; **89**: 414–20. http://www.kamalnigam.com/papers/canopy-kdd00.pdf.

6 Dunn HL. Record linkage. *American Journal of Public Health* 1946; **36**: 1412–16.

7 Newcombe HB, Kennedy JM, Axford SJ, James AP. Automatic linkage of vital records. *Science* 1959; 130: 954–59.
8 Fellegi IP, Sunter AB. A theory for record linkage. *Journal of the American Statistical Association* 1969; **64**: 1183–10.
9 Newcombe HB, Kennedy JM. Record linkage: making maximum use of the discriminating power of identifying information. *Communications of the Association for Computing Machinery* 1962; **5**: 563–67.
10 Newcombe HB. *Handbook of Record Linkage: Methods for Health and Statistical Studies, Administration, and Business*. Oxford: Oxford University Press, 1988.
11 McCallum A, Nigam K, Unger LH. Efficient clustering of high-dimensional data sets with application to reference matching. *Sixth ACM Conference of Knowledge Discovery and Data Mining*, 169–78, 2000. http://www.kamalnigam.com/papers/canopy-kdd00.pdf.
12 Baxter R, Christen P, Churches T.A Comparison of fast blocking methods for record linkage. *Proceedings of the ACM Workshop on Data Cleaning, Record Linkage and Object Identification,* Washington, DC, August 2003.
13 Winkler WE. Improved decision rules in the Fellegi-Sunter Model of record linkage. Pro*ceedings of the Section on Survey Research Methods*, *American Statistical Association* 1993; 274–79.

APPENDIX

Figure A shows a use case for the proposed system. Use-cases are structured outline or templates used for describing components of a system. They consist of actors or external agents lying outside the system model but interacting with the system in some way. An actor may be a person, machine or an information system that is external to the system model.

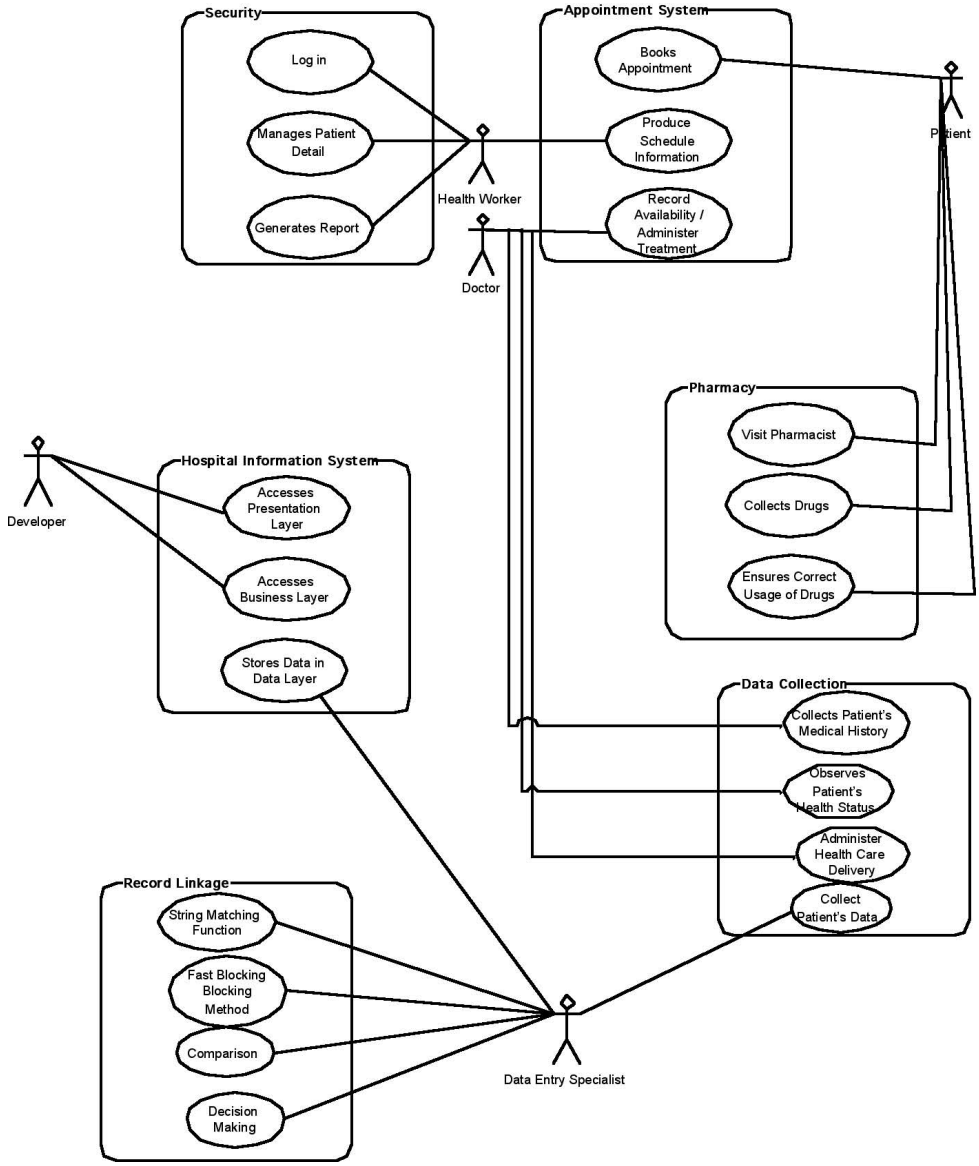

Figure A. *Use-case Diagram for the Proposed System*