MEDINFO 2001
V. Patel et al. (Eds)
Amsterdam: IOS Press

Medical Records and Electronic Documents: A Proposal

Frédérique Laforest, André Flory

Medical Informatics Research Group, Laboratoire d'ingénierie des systèmes d'information (LISI), Lyon, France

Abstract

This article presents a global view of our proposal for a medical information system of the future. This information system focuses on patient medical records management. As most of the existing systems, it proposes to store information from patient records in a database. But records capture is different: we propose to use weakly-structured documents. Such documents contain paragraphs with some specific constraints represented by XML tags. The end-user writes new information under the form of weakly-structured documents. An internal system translates theses documents into new data for the internal database. Such a document-based user interface provides much more freedom to the end-user, and certainly reduces the distance between the physicians' way of working and the capture system.

Keywords

medical record, information systems, semi-structured documents, XML, databases

Introduction

A database is a repository used to store structured data about an application domain. Relational databases are currently the most used. They represent all data as a set of tables, each column represents a sort of data (e.g. patient name, diagnosis, medication prescribed). Each column is typed so that the format of data is fixed. A database is not accessed directly, but through a software called database management system (DBMS). The DBMS authenticates users... but most of all, it allows to query data in a very precise and efficient way tanks to the standard SQL language. This is the reason why it is quite always used in computerized systems that store a lot of information. The greatest drawback of databases comes from its greatest advantage: data structure. The data model cannot be modified easily; adaptability is difficult to ensure. By the same way, as data are highly structured in storage, a lot of computerized systems ask for the end-user to follow this structure: they present form-based windows, in which the end-user has to fill fields in the right format [1]. This rigidity is the most important reason why such computerized medical records are not widely spread [2]. Some proposals prefer not using a database, so that the structure of information is not so prominent. Most of them propose to write documents.

The simplest propose to write free-text documents, which only contain information and presentation guides. In such systems, automated information searching is very difficult ([3]). Others propose to use semi-structured documents ([4,5]). Semi-structured documents include both information and meta-information, i.e. semantics of information pieces. SGML and XML ([6,7]) are specific languages that allow to insert meta-information in texts. This meta-information is represented by tags. Adaptability of these systems is much improved. Searching for information is easier than in free-text documents, but it is not as precise as in databases. A lot of research effort is currently done to improve query languages on XML documents([8,9,10,11]). Semi-structured documents offer more freedom to the end-user, but their lacks in information retrieval impedes their effective use.

As each type of system has its advantages, we propose to make a system that mixes both techniques. Our proposal uses a document-based user interface, coupled to an underlying database. Before presenting our system, this article presents different types of user interfaces. The first section discusses the classical form-based user interfaces. The second section presents the three main kinds of documents and presents which one we have chosen. The third section presents the system we propose: first we explain the user interface, and secondly we expose the architecture of the system. We conclude then with the state of our prototype and the future advances of our project.

User interfaces to database-based systems

First electronic documents date from a long time ago. They are used since the beginning of end-user interfaces. They consist of electronic representations of paper-based forms like administrative sheets, blood analysis requests... These forms contain variable areas that need to be filled by the end-user.

Computerization of activities in companies started for management domains with "numerical" and repetitive tasks

[12,13]. Stocks, orders, invoices are all domains that mainly manage numbers. They also follow precise rules that cannot be circumvented. In such domains, defining rules and information to manage is a well known process. The database model and electronic forms for the user interface are defined from information and rules.

If the use of forms is well adapted to management domains, it is not the case for much more complex domains, where human expertise is required. The medical record is such a domains. Each medical record is intrinsically unique: if some steps can look like form-based domains (e.g. biological analyses results), there is a huge variability between patients, which directly impacts the medical record structure. All experiments conducted in this domain agree on that fact [14]. This is the most important reason why computerized medical records based on forms are so heavily criticized by end-users: no predefined form can adapt to each case. Using other kinds of electronic documents could reduce the mismatch between end-users needs and medical records management in databases.

Improvement : using "true" documents

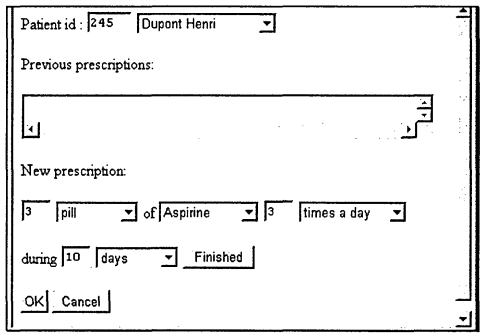
In the literature, one can find three main types of electronic documents (see fig. 1):

- Form-based documents as explained in the previous section contain only formatted information.
- Unstructured documents contain flows of information, without any additional clue on the nature of information pieces. Documents written with classical editors (like Words®) are of this kind. They are often called free-text documents.
- Semi-structured documents contain flows of information, but can also point at specific information pieces [15]. To do so, meta-information is added to the text under the form of tags, as in SGML or XML.

Unstructured and semi-structured documents provide a more natural way to capture information. Paper-based documents in the medical record look much like these documents. At the end-user's point of view, using them as a user interface paradigm would be a step forward to computerize medical records. This technique is proposed by some systems. Most of these systems only replace the pen by the keyboard, but do not offer additional functionalities, as compared to paper-based records. We found none that would fill a domain-oriented database model. The few remaining systems try to discover automatically some information pieces in the record, and use them to index records [16,17,18]. They provide an improved research mechanism as compared to paper-based records. But automatic indexing is not totally reliable as information detection is not precise. Full-text searches are also sometimes implemented: given a set of words, the system selects all documents containing the given set.

Semi-structured documents seem more adapted to the electronic management of medical records [19]. As they contain tags to locate information pieces, they can be more precise to search information: search algorithms do not search the whole document, but only tagged pieces in which they can find an answer to the query [20].

Form-based document:



Unstructured document:

```
Patient 245 Dupont Henri
Give 3 pills aspirin 3 times a day during 10
days
```

Semi-structured document in XML:

```
<patient id='245'>
<name> Dupont </name>
<first_name> Henri </first_name>
<prescription> Give <dose> 3 </dose>
<dose_unit> pills </dose_unit> of <medication
id='12'> aspirin </medication> <frequency> 3
</frequency> times a day during <duration> 10
days </duration> </prescription>
</patient>
```

Figure 1: Three kinds of documents

The use of an underlying database is today the better way to manage data. So, in this hypothesis: How to associate a document-based user interface and an underlying database?

The main proposal of this article is to use semi-structured documents to capture medical records. Parts of the captured text are stored in the database. Using semi-structured documents offers more freedom and more adaptation to the end-user than form-based documents. While forms impose the list of information to capture, an order in which it has to be done, in documents information can be captured in different orders, some data are optional (and do not appear as "empty areas" in the document), presentation rules can be different from one user to another. Resulting documents only have to follow the tags grammar, which defines which tags can be used, and some relationships between tags. XML [7] is the currently most studied language for writing semi-structured documents. The tags grammar is defined in a Document Type Definition file (DTD).

There are two sorts of semi-structured documents [21]:

- Strongly structured documents tag information very precisely. Each information item that could be queried

is tagged. On fig. 2, tags locate a medication name, or a dose, or a frequency or a duration. Such documents are today used to make system-to-system communication [22,23], but cannot be realistically used to capture medical records: writing such documents is really time consuming, and we are sure no doctor or nurse could find time to write strongly-structured documents.

- Weakly-structured documents do not tag each interesting information piece, but only locate paragraphs in documents. They provide semantics to paragraphs, so that a reader understands the context in which he should read the paragraph. For example (see fig. 2), a tag locates a prescription sentence, but no more detailed tag is added. Weakly-structured document seem more adapted to medical records capture. Paragraphs tagging is significantly less time-consuming than data tagging. Moreover, paragraph tags help much data detection : tags provide a context for information interpretation by the computer

We thus propose a system where end-users write medical records under the form of weakly-structured documents. [24] makes the same distinction as we do, and also proposes to use weakly-structured documents for medical records capture. But in our proposal, we go further: a document analyzer parses documents to induce new data to store into the database. Information capture is highly close to the paper-based medical record, and the computer can help the end-user secure capture (e.g. prescription control) and search information in the database.

Strongly-structured document tag each data:

```
<patient id='245'> <name> Dupont </name>
<first_name> Henri </first_name>

<prescription> Give <dose> 3 </dose>
<dose_unit> pills </dose_unit> of <medication
id='12'> aspirin </medication> <frequency> 3
</frequency> times a day during <duration> 10
</duration> <duration_unit> days </
duration_unit > </prescription></patient>
```

Weakly-structured document tag each paragraph:

```
<patient id='245'> <name> Dupont Henri
</name>

<prescription> Give 3 pills of aspirin 3
times a day during 10 days</prescription>

</patient>
```

Figure 2: Strongly-structured and weakly-structured documents (containing the same information)

Architecture of a document-based system

Captured documents are stored as is, but they are also read by a document analyzer. It extracts data from documents and sends them to a relational database. Finding information about a patient, or making statistical studies on information is made with the database. The database ensures that answers are complete and entirely exact.

Short description of the user interface

Entering the software to capture medical records, the doctor is asked for identification and password. He thus selects a patient in the list of patients of his department, and the type of the document he wants to write. The document type can be an analysis request, an encounter resume... The document type allows the system to know which set of tags can be inserted in the document. The document capture window is then presented, as shown on fig. 3.

This window contains mainly two parts:

- On the left side, one can find the paragraphs structure of the document: it is a tree structure. Each paragraph type is represented as a new node in the tree, the text of the paragraph is presented as a leaf. As our system only allows to tag paragraphs, the tree is limited to depth 2 (at depth 1: document identification information, at depth 2: the paragraph tags selected by the doctor).
- On the right side, the text written by the doctor. The text is typed totally freely, as in free-text editors.

The first paragraphs including patient id and date are automatically filled by the system, according to the selections previously made by the doctor.

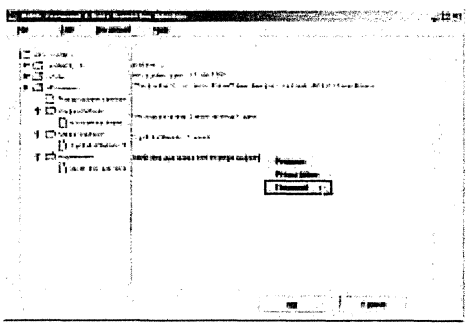


Figure 3: Patient record document capture window

The doctor writes paragraphs in the right area as he would do in a free-text editor. When he finishes writing a paragraph, he right-clicks on the paragraph, and a list of possible tags is proposed. He selects the corresponding one, and the left-side of the window is updated : a new paragraph is added to the tree.

If necessary, the doctor can use the "Edit" menu to select information from a classification or a thesaurus. For example, he wants to write a diagnosis from the ICD-10. He selects the ICD-10 sub-menu, the classification is shown. The term he selects is added at the location of the cursor.

When the doctor clicks on the "OK" button, the document is stored as is in the patient record directory. It is also sent to the document analyzer to extract information. Extracted information is presented to the end-user for validation. The document analyzer sends validated data to the database.

Architecture of the system

Fig. 4 presents the main components of our system.

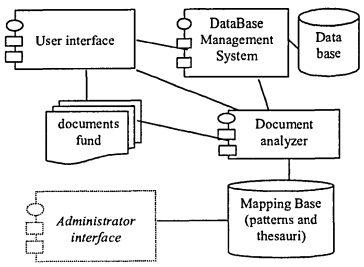


Figure 4: Architecture of our system

- The user interface

As shown above, the user interface permits the capture of documents under the form of weakly-structured documents. It also manages the validation of extracted information. The end-user verifies extraction is correct and complete. The user interface also permits querying the database. It may have two objectives: select documents or work directly on data (temperature diagram, statistical analyses...).

- The document fund

It contains all the documents written about all patients. They are the direct representation of the paper-based medical record. They are the legal source of information.

- The database

The medical record database contains fields usually present in electronic medical record systems that use form-based documents. We just add documents references columns.

- The database management system

The DBMS is a commercial software. It manages all communications with the database. It gets requests to the database, verifies the user is allowed to do so, makes modifications on the database according to queries and returns results if any.

- The document analyzer

The document analyzer gets filled documents written by doctors or nurses and analyzes them to extract information. It extracts information that have a corresponding field in the database. Document analysis uses a mapping base.

- The mapping base

The mapping base contains rules to define how to identify data in paragraphs, and how to send data to the database. Identification of information in paragraphs is based on pattern matching, i.e. on the detection of classical ways to write parts of sentences. These patterns allow to identify data.In a prescription paragraph, dosage is often written like this : "take 3 pills of ...". This classical form allows to identify the dosage value and the dosage unit. Some parts of the patterns to discover are directly searched in classifications or thesauri, e.g. medication names.

Rules are defined at three levels :

- The rule level is at the upper level. It divides each paragraph into a set of segments, without any constraint on segments appearance order.
- The segment level divides each segment into a set of alternative expressions that could be written for it.
- The expression level is the most precise level. It provides the ordered list of words that should appear in an expression. It also provides the place each term should take in the database.

Fig. 5 provides a formal description of rules grammar. Fig. 6 shows an example on prescription. The PrescriptionRule rule explains that a prescription should contain an optional dose segment, an optional duration segment and a compulsory drug segment. The durationSeg rule explains that there are 2 ways to write a duration : durationExpr1 or durationExpr2. The durationExpr1 rule explains that the system should find (1) a term coming from the duringWordsList ("during", "for"...), followed by (2) a number that should be stored into the duration column of the Posology table in the database, followed by (3) a term coming from the UnitsList and that should be stored into the durUnit column of the Posology Table.

```
rule : ruleName '=' segment option* [',' segment option*]*

segment : segmentName '=' expression ['|' expression]*

expression : exprName '='
        (thesaurus':'table.attribut)|thesaurus
        [',' (thesaurus':'table.attribut)|thesaurus]*

option: '?'        //for optional segments
```

Figure 5: Rules grammar

```
PrescriptionRule = doseSeg ?, durationSeg?, drugSeg

durationSeg = durationExpr1|durationExpr2

durationExpr1= duringWordsList,
      NbList:Posology.duration, UnitsList:Posology.durUnit
```

Figure 6: An example of each rule level

Writing rules with 3 levels has two main advantages. Rules are understandable: each level is easy to read, and libraries of expressions and segments can be made for reuse.

Sending data to the database requires writing SQL queries. Some translations are necessary to transform discovered data into an adequate format to the database fields. SQL queries are written depending on discovered data.

- The administrator interface

The administrator interface is used by computer scientists to update the mapping base. This task includes removing, adding or updating thesauri and classifications. It also allows to adapt existing rules or to write new rules.

Conclusion

We have developed a prototype of the analyzer for medication prescriptions which proved to be efficient. It is written in Java and uses the DOM model to manipulate

documents [25]. We are extending this work to all paragraphs that could appear in a patient medical record. The end-user interface is developed in Java, in a distributed environment using Enterprise Java Beans [26].

We are also working on improvement of the rules internal format: the studies proved that our rules have to be improved. We now study a complete transformation process including 4 main steps, each concerning a different kind of rule: selection rules to select efficiently tagged paragraphs; extraction rules to extract relevant information from paragraphs; processing rules to format extracted information; structuring rules to build SQL queries.

References

[1] Stair T.O Reduction of redundant laboratory orders by access to computerized patient records. *J. Emerg. Med* 1998;16:895-7

[2] Berg M. Medical work and the computer-base patient record : a sociological perspective. *J. Methods Inf. Med.* 1998;37:294-301

[3] Salton G. Another look at automatic text-retrieval systems. *Communications of the ACM*, july 1986, 29(7): 648-656,

[4] Charlet J. & al. Hospitexte: towards a document-based hypertextual electronic medical record. *Journal of the AMIA*, 5(suppl):713-717, 1998.

[5] Bouaud J., Séroussi B., & Zweigenbaum P. An experiment towards a document-centered hypertextual computerised patient record. Brender J. & al. eds, *Proceedings of MIE'96*, Copenhagen: IOS Press 1996: 453-457

[6] ISO, Information processing - Text and office systems. Standard Generalized Markup Language (SGML), *ISO 8879*, 1986.

[7] World Wide Web Consortium. *Extensible Markup Language(XML).* http://www.w3.org/XML/#9802xml10

[8] Buneman P., Davidson S. and Hillebrand G. A query language and optimization techniques for unstructured data. *ACM SIGMOD'96*: 505-16

[9] Deutsch A. et al. A Query Language for XML. *Int. WWW Conference*, 1999

[10] Christophides V. et al. From structured documents to Novel query facilities. *ACM SIGMOD Conference*, may, 1994

[11] Cluet S., Deutsch A. et al. *XML query languages : experiences and exemplars.* http://www-db.research.bell-labs.com/users/simeon/xquery.html

[12] De Moor G.J.E. The promise of medical informatics in Europe. Van Bemmel J.H., McCray A.T. Eds. *Yearbook of medical informatics*, 1999. New-York: Shattauer. 58-61

[13] Borst F. & al. Happy birthday DIOGENE: a hospital information system born 20 years ago. Cesnik B. et al. Eds. *MedInfo'98.* Amsterdam: IOS Press, 1998:922-6

[14] Nordyke R.A., Kulikowski C.A. An informatics-based chronic disease practice: case study of a 35-year computer-based longitudinal record system. *J. AMIA.* 1998;5:88-103

[15] J. McHugh, S. Abiteboul and al. Lore : a database management system for semistructured data. *SIGMOD Records*, 26(3):54-66, septembre 1997

[16] Zweigenbaum P. & al.. MENELAS: an access system for medical records using natural language. *Computer Methods and Programs in Biomedicine*, 1994;45:117-120

[17] Romacker M. et al. Semantic analysus of medical free texts. Hasman A. et al eds. *MIE'98.* IOS Press, 2000:438-42

[18] Ruch P. et al. Tagging medical text : a rule-based experiment. Hasman A. et al eds.*MIE'98.* IOS Press, 2000:448-55

[19] Frénot S, Laforest F. Medical record management systems : criticism and new perspectives. *Methods of Information in Medicine* 1999; 38:89-95

[20] Le Maitre J., Murisasco E. Rolbert M. From annotated Corpora to databases : the SgmlQL language. J. Narbonne Ed. *Linguistic Databases*, CSLI Lecture Notes. 1998; 77: 37-58.

[21] Bourret R. XML and Databases, September 2000 http://www.rpbourret.com/xml/XMLAndDatabases.htm,

[22] Kahn C.E. Self-documenting structured reports using open information standards. Cesnik B. et al. Eds. *MedInfo'98.* Amsterdam: IOS Press, 1998:403-7

[23] Kooijman C.J., Kaag M.E.C. Sending specialist reports to GPs using EDI. Cesnik B. et al. Eds. *MedInfo'98.* Amsterdam: IOS Press, 1998:408-11

[24] Tange H.J., et al. The granulariy of medical narratives and its effect o,n the speed and completeness of information retrieval. *J. Am. Med. Inform. Assoc.* 1998;5:571-82

[25] W3C *Document Object Model (DOM)* http://www.w3.org/DOM/

[26] Sun *Enterprise Java Beans technology* http://java.sun.com/products/ejb/

Address for correspondence

Frédérique Laforest, André Flory

LISI, INSA Lyon, bat Blaise Pascal, 69621 Villeurbanne Cedex, France

33 (0) 4 72 43 89 83

{frederique.laforest, andre.flory}@insa-lyon.fr