Towards a Prototype Medical System for Devices Vigilance and Patient Safety

Antonios Deligiannakis
School of Electronic and Computer Engineering
Technical University of Crete
Chania, Greece
adeli@softnet.tuc.gr

Nikos Giatrakos
School of Electronic and Computer Engineering
Technical University of Crete
Chania, Greece
ngiatrakos@softnet.tuc.gr

Nicolas Pallikarakis
Biomedical Technology Unit
University of Patras
Patras, Greece
nipa@upatras.gr

***Abstract*—For all healthcare institutions and organizations, patient safety is of the utmost importance. A factor that influences patient safety is the existence (or not) of observed adverse events associated with medical devices. Upon the detection of adverse events, all healthcare providers that own the affected medical devices should be promptly notified. In this paper we present the core of a prototype system for medical devices vigilance and patient safety. We present the architecture of this system, the way that it detects the healthcare providers that need to be notified through an entity matching algorithm, as well as briefly present its user interface.**

I. Introduction

For all healthcare institutions and organizations, patient safety is of the utmost importance. The World Health Organization (WHO) identifies patient safety as a serious global public health issue. According to WHO, patient safety is defined as the prevention of errors and adverse effects to patients associated with health care. While most patient safety issues are associated with the acquisition of infections, some errors correspond to *adverse events/incidents* related to medical devices. An event is classified as adverse if it causes, or may potentially cause, unexpected or unwanted effects involving the safety of (primarily) patients, users or other persons. Examples of such adverse events are misconnections in ECG cables that may result in electrocution, fires during defibrillation (typically caused by insufficient paddle contact, with subsequent arcing, in the presence of oxygen-enriched environments), or even errors that may occur, due to a complex user interface, by the machine operator. While medical devices are produced and distributed according to international standards, and have the necessary certification in compliance with the Medical Devices Directives and guidelines for the EU, and/or FDA approval for the US, in clinical practice even the best designed products, could potentially fail and put patient and/or staff safety at hazard.

The importance of reporting adverse events is evident from the large number of organizations that report them. Some examples include, but are not limited to: (1) FDA [1] - U.S. Food and Drug Administration, (2) ECRI [2] - Emergency Care Research Institute, (3) TGA [3] - Australian Department of Health, (4) IMB [4] - Irish Medicine Board, (5) SwissMedic [5] - Swiss Agency for therapeutic products, (6) EUDAMED [6] - European databank for medical devices, (7) MHRA [7] - Medicines and Healthcare products Regulatory Agency, (8) MEDSAFE [8] - New Zealand Medicines and Medical Devices Safety Authority etc.

Once an adverse event occurs, healthcare providers that possess the medical device related to the adverse event should be promptly notified, in order to take appropriate measures. The common practice is that the manufacturer of the relevant medical device should contact all healthcare providers that possess the medical device at question. However, this is not always done, either due to unwillingness of the company, or due to the large number of involved health providers, which is difficult to track, especially at a global level. What instead seems as a more reliable solution would be to establish a medical devices vigilance system that would assume the role of collecting information about adverse events and then notify all appropriate healthcare providers, assuming that they have registered the medical devices that they possess. To avoid such adverse events or prevent their repetition, the European Directives on medical devices include provision for the establishment and operation of medical device vigilance systems in the European Union [9], [10]. These systems must be created under the responsibility of each member state and should collect reports of adverse events involving medical devices, perform investigation when appropriate, and disclose the information to the other member states and the EU Commission in order that necessary precautions be taken [11], [12]. However, countries have not yet implemented such a system and many adverse events remain unknown.

In this paper we present the database component and the web interface of the MEdical DEvices VIgilance and PAtient Safety (MEDEVIPAS) that we are currently developing. While the goal of the system is to be deployed at Greek hospitals and its current web interface is in English, the design of its database engine has been performed in a way that allows its operation even when the registered information is in different languages. This is achieved through the use of aliases, with which the user can specify that particular keywords (in different languages) correspond to the same entity.

The contributions of this paper are summarized as follows:

- We propose MEDEVIPAS, a novel system for medical devices vigilance and patient safety. The goal of the system is to automatically inform healthcare providers for adverse events related to medical devices that they possess, thus enhancing patient safety.

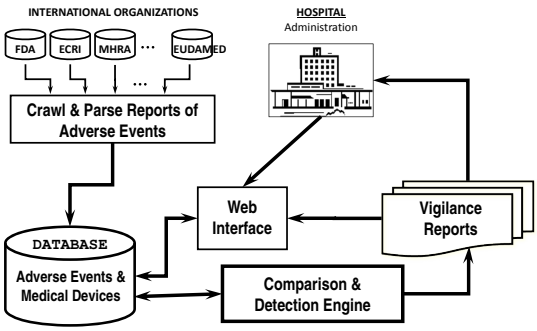


Fig. 1. Architecture of MEDEVIPAS

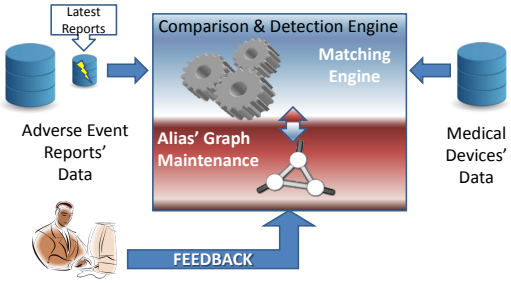


Fig. 2. Vigilance Prototype's Building Blocks

- We present our algorithm for *entity matching*, which helps determine, based on newly reported adverse events, which medical devices may be influenced.
- We present enhancements to our design based on the notion of aliases. We explain how aliases can help our entity detection algorithm in the case of alternative spellings for the same entity (i.e., Siemens and Ziemens are both valid names of the same company in different languages). We demonstrate how aliases can be efficiently maintained based on an alias graph.
- We present a detailed view of the database organization, as well as the web interface of MEDEVIPAS.

Roadmap. This paper proceeds as follows. In Section II we present the overall architecture of MEDEVIPAS. In Section III we present the data infrastructure of our system, including basic components of the stored data and the description of our alias graph. In Section IV we present our matching algorithm. In Section V we present our web application, while Section VI presents concluding remarks and future directions.

II. MEDEVIPAS Architecture

We now present the overall architecture, depicted in Figure 1, of the MEDEVIPAS system that we are currently developing. The system needs to collect information regarding adverse events. In order to achieve this, crawlers need to be developed for the international authorities that publish such information, such as FDA [1], ECRI [2], MHRA [7], TGA [3], IMB [4], SwissMedic [5], etc.

We need to note that the format of the reported adverse events in all of these resources, while being fairly well structured, is not uniform, and all resources do not contain exactly the same fields of information. On one hand, the variety at the format of the reported data makes it hard to distinguish cases when the same adverse event is report in more than one authority (i.e., by both FDA and ECRI at different times). Our system will treat these two reports as two different detected adverse events, thus potentially notifying healthcare authorities that possess the corresponding medical device more than once. On the other hand, a decision needs to be made on whether the data regarding the crawled adverse events will be stored in multiple database tables (i.e., one per crawled authority), or at a common format, which contains fields that are mentioned in many of these sources. We opted for the second option, as (1) it simplifies the design of the database, and (2) the url for each crawled adverse report can be stored in the database, thus allowing the user to view any fields of information not stored in the database. In particular, in Section III we present the fields maintained for each adverse event, with these fields best matching the format of FDA, which typically reports more adverse events, compared to the other listed authorities.

Besides data relevant to adverse events, the MEDEVIPAS database also maintains information about medical devices that healthcare authorities have registered in the system through the web interface of the system. Periodically (currently this is performed every 24 hours), if new tuples have been inserted in the database, our entity matching algorithm (discussed in Section IV) is executed. If new matches are obtained, the healthcare authorities with the matched medical devices are notified, receiving a vigilance report.

III. Data Infrastructure

We set out the presentation of our prototype by describing its basic infrastructure. We focus on the corresponding building blocks of our vigilance system of Figure 1 and present them in more detail in Figure 2. An overview of the corresponding infrastructure is depicted in the class diagram of Figure 3 and will be analyzed in this section.

Medical Devices Data. Hospital medical device inventory data (on the right of Fig. 2) is created and maintained by regular data import from existing hospital devices inventories. In its current version the MEDEVIPAS vigilance system supports both individual medical device insertion and batch uploading of them. For the latter case, comma separated values (CSV) files should be uploaded by a user belonging to a specific health care institution. Future versions will support uploading delimited text files or excel files. As depicted in Figure 3, the information currently stored for each medical device includes information (institution, department, location) regarding both the healthcare provider owning the device, as well as information helping identify the device - its manufacturer, model name, serial, device type in English, as well as the Universal Medical Device Nomenclature System (UMDNS) code specifying the encoding of the device type.

Adverse Event Reports' Data. The Adverse Events data (on the left of Fig. 2) can be populated with vigilance data provided mainly by international sources, such as international authorities [1-8], International Organizations, manufacturers

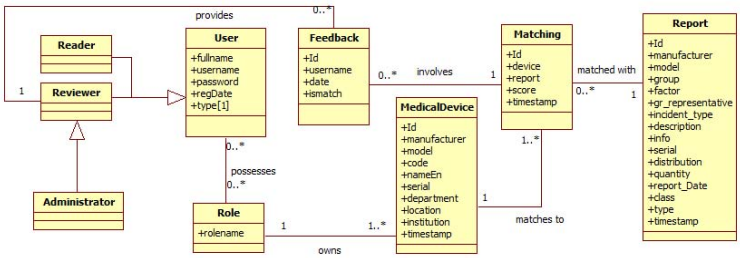


Fig. 3. Vigilance System's Data Infrastructure

and suppliers of medical equipment.[1] The data attributes stored for each adverse event include the manufacturer, model, serial and group of the involved medical device (as well as information on where it is distributed), information about the incident and the quantity of reported devices with defect, the date of the report etc.

Comparison & Detection Engine. The vigilance system includes a flexible Comparison and Detection Engine (middle component in Figure 2) that identifies and reveals potentially hazardous medical devices. The engine's operation is supported by a Matching Engine (sub)component which includes an efficient algorithm for potentially hazardous device identification. The second (sub)component involves an Alias Graph that provides the flexibility of declaring synonyms on the fields examined by the matching algorithm.

- **Matching Engine.** To pinpoint potentially hazardous medical devices, we design a matching algorithm (described in Section IV) that is capable of correlating the data between the stored adverse events and medical devices. We determined that the triplet <manufacturer, model, serial> can uniquely identify a medical device. Thus, our algorithm for correlation and detection is based on flexible comparison of the triplet <manufacturer, model, serial> of the fields of the medical devices and reports' data. The output that is stored in the database consists of detected potentially dangerous medical devices along with corresponding adverse event reports and their matching score.

- **Alias Graph.** The declaration of entity aliases is facilitated by the construction and maintenance of an Alias Graph component. To conceive the utility of entity alias, consider for instance a specific manufacturer entity $Manuf_A$. Some of the reports of adverse events may often use the terms $Manuf_A\ International$ and $Manuf_A$ interchangeably. This example expresses the need for synonym declaration for the same manufacturer entity. As a result, users should have the opportunity to register such common synonyms in the database for the matching algorithm to take into consideration, since string comparison operations under any function may output low matching score for the above instance when trying to correlate a specific report towards a particular piece of medical equipment in one of the registered healthcare institutions. Synonym declaration also provides us with extra flexibility, which is required for handling multilingual data, i.e., when inserting into the database data that are extracted from data sources of different language, regarding reports of adverse events.

An alias can be declared on either individual components (i.e., manufacturer name, or model name) of inserted data, or on combinations of components (i.e., the combination of <ManufacturerA, ModelA> and <ManufacturerB, ModelB> are equivalent - note it is likely that the same product be distributed under two completely different manufacturer and model names, in the case of companies that have been bought by the same group). Upon annotating an alias for a particular entity, our prototype retains the flexibility of automatically detecting the existence of transitive (closure) dependencies through the self-maintained Alias Graph Component. The Alias Graph component is stored as a simple, double columned database table of correspondences. Whenever a correspondence (alias) is specified between two entities EntityA and EntityB, two tuples are inserted in the table of correspondences: tuple <EntityA,EntityB> and tuple <EntityB,EntityA>. The reverse operation is performed upon the deletion of an entity correspondence (i.e., due to an error). Then, using a recursive procedure, one can create/update the alias graph, which stores all pairs of entities that are connected and are, thus, considered to be equivalent. Schematically, the aforementioned recursive procedure can be expressed in the form of the following query (for database systems allowing WITH RECURSIVE clauses), issued upon a new entity's Y in a <X,Y> pair insertion:

```
WITH RECURSIVE transitive_closure
(EntityA, EntityB, distance, path_string) AS
( SELECT EntityA, EntityB, 1 AS distance,
EntityA + '.' + EntityB + '.' AS path_string
FROM AliasGraph
WHERE EntityA = 'X'
– set the starting node, existing entity 'X'
UNION ALL
SELECT tc.EntityA, ag.EntityB, tc.distance + 1,
tc.path_string + ag.EntityB + '.' AS path_string
FROM AliasGraph AS ag
JOIN transitive_closure AS tc ON ag.EntityA = tc.EntityB
WHERE tc.path_string NOT LIKE '%' + ag.EntityB + '.%'
)
INSERT INTO AliasGraph VALUES
SELECT 'Y',EntityB FROM transitive_closure;
–maintain the graph for the new entity 'Y'
```

Using this Alias Graph Component, every time that a new alias is declared, the Comparison and Detection Engine (Section IV) updates its estimated scores of matching certain

[1]Currently, data crawled from FDA and TGA are stored, but crawlers for the other data sources are under development.

adverse events against medical devices. Obviously, entity instances that do not appear to be linked in the aforementioned graph have their cohesion determined by the Matching Engine Component.

Feedback. Using the invented algorithm, the Matching Engine Component suggests possible matching relations, which undergo approval by a user which reviews and validates or not extracted outcomes (feedback). The procedure aims to provide continuously improving vigilance results. Customized vigilance reports are generated, and after being further assessed and verified via the feedback component, are distributed to the health care institutions concerned, in order to take the necessary preventive actions.

The data infrastructure presented so far is depicted in the class diagram of Figure 3. To keep the diagram readable, the double columned table of Alias Graph maintenance has been omitted, but its information content and connection to the medical devices as well as adverse event instances has already been presented. The scope of classes regarding the specification (ISA) on the User class as well as the Role class will be discussed in Section V.

IV. Our Matching Algorithm

In the current section we elaborate on the entity matching algorithm which is the core of the Comparison and Detection Engine of our prototype. The whole process is sketched in Algorithms 1, 2.

In Lines 1-7 of Algorithm 1, we examine whether in previous runs of the matching process, if any, we have already matched certain tuples in the Medical Devices dataset D with Adverse Events' reports in R. To do so, we take advantage of the timestamp attribute (see Figure 3). In particular, $lastTime$ stores the last execution of the matching algorithm. If this is not the first execution of the algorithm, we only consider the newly inserted (i.e., inserted after $lastTime$) reports and new medical devices' inventory data in the $R' \subset R$ and $D' \subset D$ relations, respectively. On the first execution of the algorithm, $R' \equiv R$ and $D' \equiv D$. This test at the beginning of the matching process enables the incremental execution of the algorithm, which can result in a dramatic reduction in its execution time, since we expect only small subsets R', D' of D, R to be examined for matching.

In Lines 8-16, the algorithm checks for matching each new adverse event report with the whole collection of medical devices inventory data. Each time the matching score extracted by the $MatchScore$ function (Algorithm 2) exceeds a given threshold parameter θ, a new matching pair is inserted in M. The details on computing the matching score are provided shortly. Note that in the first execution of the algorithm, the whole collection of medical devices is checked for matching against the available reports. We then need to examine (Lines 17-25) for matching the latest medical devices inserted after the last execution of the algorithm, making sure that they are not matched against reports in R', since this matching has already been performed in Lines 8-16. Finally, in Line 26 we update the timestamp $lastTime$.

Algorithm 2 presents the function according to which the matching score for each candidate pair (ξ_i, ξ_j) is computed.

Algorithm 1 Entity Matching Algorithm

Input: (a) Medical Devices' Data D,
(b) Adverse Events' Reports Data R,
(c) Matched Data M: (ξ_i, ξ_j, $Score$, t). Stores score of matching ξ_i with ξ_j at time t,
(d) Alias Graph G,
(e) Weights for components of the matched triplet: $0 \leq w_{man}$,w_{mod},$w_{ser} \leq 1$
(f) Matching threshold θ

Ensure: Updates M with new matches $\{\xi_i \longleftrightarrow \xi_j\}$, $\xi_i =< manuf, model, serial, t >$

1: /* Store in R', D' tuples that arrived after the last matched result. $lastTime$ stores the last execution of this algorithm. */
2: **if** Not first execution of matching algorithm **then**
3: $R' \leftarrow \{\xi_i \in R : \xi_i.t > lastTime\}$
4: $D' \leftarrow \{\xi_i \in D : \xi_i.t > lastTime\}$
5: **else**
6: $R' \equiv R$, $D' \equiv D$
7: **end if**
8: // Next, check only new reports against all listed devices
9: **for all** $\xi_i s \in R'$ **do**
10: **for all** $\xi_j s \in D$ **do**
11: $Score \leftarrow$ MatchScore($(\xi_i$,$\xi_j)$, G, w_{man},w_{mod},w_{ser})
12: **if** $Score \geq \theta$ **then**
13: $M \cup (\xi_i$,$\xi_j, Score, t_{now})$
14: **end if**
15: **end for**
16: **end for**
17: // Next, check only new devices against unmatched reports
18: **for all** $\xi_i \in D'$ **do**
19: **for all** $\xi_j \in R \setminus R'$ **do**
20: $Score \leftarrow$ MatchScore($(\xi_i$,$\xi_j)$, G, w_{man},w_{mod},w_{ser})
21: **if** $Score \geq \theta$ **then**
22: $M \cup (\xi_i$,$\xi_j, Score, t_{now})$
23: **end if**
24: **end for**
25: **end for**
26: Update value of $lastTime$

In Lines 1-3 of the function, we first examine whether the two pairs of attributes $< manufacturer, model >$ appear as aliases in G, in which case both the respective similarity values are set to 1. If this is not the case, in Lines 4-6, we check if just the manufacturer attribute of the candidate pair is declared as a synonym in the Alias Graph G. In that, the similarity of the corresponding manufacturer attribute is set to 1 (Line 5) and only the similarity of the model attribute is computed (Line 6). If the examined pair (ξ_i, ξ_j) does not possess attributes declared in G, both the model and the manufacturer similarity values are computed (Lines 7-10). Eventually, the similarity of the serial attribute is computed (Line 11) and individual similarity scores are normalized (Lines 12-14), before the overall Matching Score is calculated (Line 15) as the weighted average of the (normalized) similarity values among the triplets of attributes. w_{man},w_{mod},w_{ser} involve the weight placed on the manufacturer, model and serial attributes, respectively, enabling users to adjust/reorder the final matches by (optionally) emphasizing in one attribute or the other. The final matching score is returned at Line 16.

Algorithm 2 MatchScore Function

Input: (a) Candidate pair (ξ_i,ξ_j),
(b) Alias Graph G,
(c) $0 \leq w_{man},w_{mod},w_{ser} \leq 1$

if ($<\xi_i.manuf, \xi_i.model>$, $<\xi_j.manuf, \xi_j.model>$) are connected in G **then**
  $Sim_{man} \leftarrow 1$
  $Sim_{mod} \leftarrow 1$
else if $(\xi_i.manuf, \xi_j.manuf)$ are connected in G **then**
  $Sim_{man} \leftarrow 1$
  $Sim_{mod} \leftarrow d_w(\xi_i.model, \xi_j.model)$
else
  $Sim_{man} \leftarrow d_w(\xi_i.manuf, \xi_j.manuf)$
  $Sim_{model} \leftarrow d_w(\xi_i.model, \xi_j.model)$
end if
$Sim_{ser} \leftarrow d_w(\xi_i.serial, \xi_j.serial)$
$Sim_{man} \leftarrow (Sim_{man} - 0.5) * 2$
$Sim_{mod} \leftarrow (Sim_{mod} - 0.5) * 2$
$Sim_{ser} \leftarrow (Sim_{ser} - 0.5) * 2$
$Score \leftarrow \frac{w_{man} \cdot Sim_{man} + w_{mod} \cdot Sim_{mod} + w_{ser} \cdot Sim_{ser}}{w_{man} + w_{mod} + w_{ser}}$
return $Score$

Algorithm 2 determines the similarity of the strings of individual attributes, whenever they are not annotated in G, utilizing the Jaro– Winkler distance d_w [13], [14]. Given two strings s_1 and s_2, their Jaro–Winkler distance d_w is:

$$d_w = d_j + (\ell \cdot p \cdot (1 - d_{Jaro}))$$

where

$$d_{Jaro} = \left\{ \begin{array}{cc} 0 & m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-q}{m}\right) & otherwise \end{array} \right\}$$

.

Jaro–Winkler distance uses a prefix scale p which gives more favorable ratings to strings that match from the beginning for a set prefix length ℓ. The standard value for this constant is $p = 0.1$ which ensures each of Sim_{man}, Sim_{mod}, Sim_{ser} for the manufacturer, model and serial attribute to yield a value lower than 1. ℓ is the length of common prefix at the start of the string up to a maximum of 4 characters.

In Jaro's distance calculation formula, m is the number of matching characters and q is half the number of transpositions. Two characters from s_1 and s_2 respectively, are considered matching only if they are the same and not farther than $\left\lfloor \frac{\max(|s_1|,|s_2|)}{2} \right\rfloor - 1$. Each character of s_1 is compared with all its matching characters in s_2. The number of matching (but different sequence order) characters divided by 2 defines the number of transpositions.

In Figure 4 we present instances of matches extracted by Algorithm 1, utilizing FDA's adverse event reports and medical devices inventory data available by the Greek Health Authorities. Results are exported as an extensible Markup Language (XML) format file.

V. Web Application

A. Managing User Permissions

In the design of our prototype vigilance system special care has been taken so as to handle and appropriately attribute user permissions. The permissions provided to a registered user can be divided in large part to two categories: (a) permissions on using specific components of the web-based application front-end (Section V-B) by introducing the notion of user types i.e., the type of a user specifies which pages and operations of the user interface are available to the user and b) restrictions on accessing certain portions of the data through the assignment of Roles based on a users' affiliation with certain hospitals or other health care institutions.

Regarding the first category, requirement analysis led us in identifying three classes of user types, namely reader, reviewer and administrator type (on the left of Figure 3). The permissions inherited by each type are:

– **Reader.** As soon as a user is registered into the system, by default inherits the Reader type. Readers can only view (with appropriate permissions) medical devices, reports and results of matches.

– **Reviewer.** Reviewers retain the permissions of a Reader user, but can also vote in order to approve/decline correspondences between medical devices and adverse events extracted by the Matching Engine. This becomes explicit in Figure 3 via the "provides" relation among the Reviewer and Feedback classes.

– **Administrator.** Administrative users possess the permissions of a Reviewer user. Moreover, a variety of system operations are available though an administrative console. Those include full manipulation of the information stored in MEDEVIPAS database, cross-checking the correctness of adverse event reports that have been automatically inserted in the database by web crawlers and approval of entity aliases proposed by reviewers.

As discussed above, the assignment of a specific type to a user restricts their ability to execute certain system operations. On the other hand, the adoption of Roles and their assignment to the users imposes constraints on the portions of data that are accessible by them. Roles essentially involve the institution (e.g., hospital or broader health care stakeholders) to which the user serves. Consequently, a Reader being a staff member of a certain hospital can only view the medical devices belonging to the inventory of that particular hospital and the corresponding adverse event reports. Similarly, a Reviewer can only provide votes on matching results where medical equipment attributed to their serving health care stakeholder is taking part.

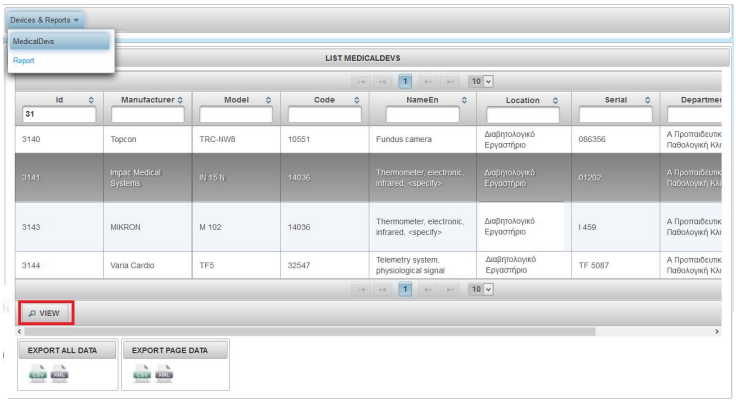
B. Web Application Components

We now provide a compact picture of the web interface of our prototype medical devices vigilance and patient safety system.

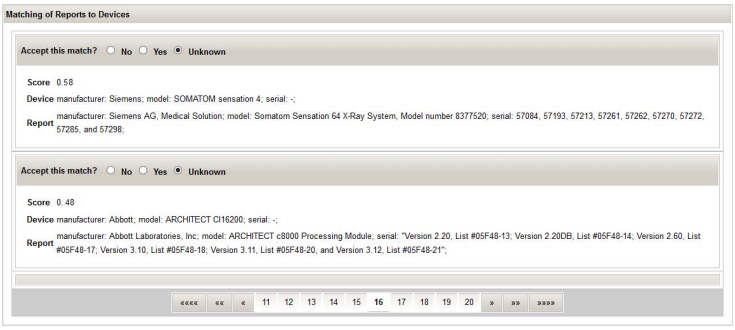
Reader Console. The Reader's web console provides mere users of the system with basic operations such as viewing records of medical devices, reports of adverse events and

```
<?xml version="1.0"?>
<datalist>
  <item>
    <score>1.0</score>
    <Device>
      <manufacturer>Beckman Coulter</manufacturer>
      <model>CYTOMICS FC 500</model>
      <serial>-</serial>
    </Device>
    <Report>
      <manufacturer>Beckman Coulter </manufacturer>
      <model>Cytomics FC 500 </model>
      <serial>All Software Versions</serial>
    </Report>
    <id>1</id>
  </item>
  <item>
    <score>0.98</score>
    <Device>
      <manufacturer>Advanced Sterilization Prods</manufacturer>
      <model>STERRAD 100S </model>
      <serial>SCZ07200027GA</serial>
    </Device>
    <Report>
      <manufacturer>Adv.Ster. Products</manufacturer>
      <model>STERRAD 100S </model>
      <serial>All serial numbers</serial>
    </Report>
    <id>2</id>
  </item>
  <item>
    <score>0.97</score>
    <Device>
      <manufacturer>Beckman Coulter</manufacturer>
      <model>CYTOMICS FC 500</model>
      <serial>ver. 2.*</serial>
    </Device>
    <Report>
      <manufacturer>Beckman Coulter, Inc</manufacturer>
      <model>Cytomics FC 500 </model>
      <serial>Software version 2.2</serial>
    </Report>
    <id>3</id>
  </item>
  <item>
    <score>0.92</score>
    <Device>
      <manufacturer>St Jude Medical CRMD</manufacturer>
      <model>EPIC II HF</model>
      <serial>111</serial>
    </Device>
    <Report>
      <manufacturer>St Jude Medical CRMD</manufacturer>
      <model>EPIC II+ HF Model</model>
      <serial>100, 106, 107, 111</serial>
    </Report>
    <id>4</id>
  </item>
```

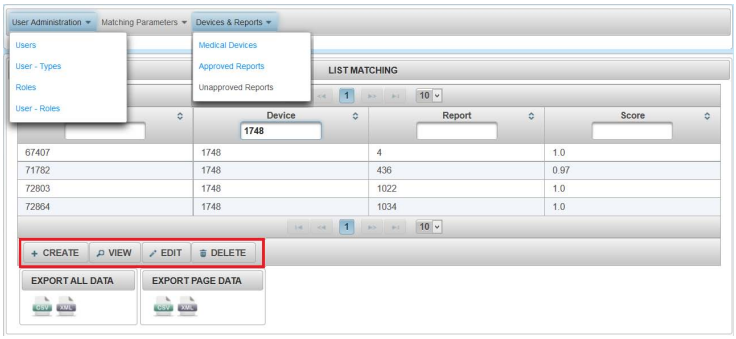
Fig. 4. XML Export (partial) of Matching Results of Algorithm 1



(a) Reader Console



(b) Reviewer Console



(c) CRUD Console

Fig. 5. Prototype Medical Devices Vigilance and Patient Safety System's Web Console

matching results accessible to them based on their role(s). Flexible searching capabilities have been implemented using filters on one or more of the corresponding data table fields. Furthermore, exports of viewed data are currently supported in CSV and XML formats. Figure 5a provides a screen-shot of the Reader's Console when viewing medical devices equipment that has been filtered on the prefix of the "Id" attribute. The menu at the top left side of the figure enables users switching among medical devices and adverse report data items while the "View" button at the bottom left corner of the screen enables users projecting the selected tuple.

Reviewer Console. The Reviewer's Console implements the feedback capabilities discussed in Section III. Through the web interface reviewers/administrators can view the results of the Matching Engine's operation which are listed as pairs of <manufacturer, model, serial> triplets for a piece of medical equipment and the matched adverse event report. As shown in Figure 5b, upon reviewing a matched pair, "Yes" (accept), "No" (reject) options are provided materializing the feedback process. In addition, the option "Unknown" is present as the default choice for pairs that have not been examined or their validity remains unclear to judge. Filtering capabilities (not shown in Fig. 5b) on the matching score are available to project only a subset of highly similar or most controversial pairs. Moreover, filters on adverse reports' publication date can be applied for disguising matches involving obsolete records.

CRUD Console. The CRUD console is available only to administrative users and serves as the front-end between an administrator and database tables. It provides the ability to execute any of the Create/Read/Update/Delete (CRUD) operations on a tuple of any database table (corresponding to classes of Fig. 3). For instance, through this console administrators can

access the feedback information provided by reviewers and approve/decline their evaluation by updating or deleting their scores.

The menu at the top left side of Figure 5c enables administrators to switch among available data. Menu items have been grouped based on whether they involve (a) user data (types, roles and their assignment) manipulation (leftmost menu items), (b) controlling the parameters of the matching algorithm, administering matching results' tuples, handling the Alias Graph and feedback information (central menu items listing matching results filtered on a specific device) or (c) applying CRUD operations (e.g. approving a newly inserted report) on medical devices and adverse event data items. CRUD operations are applied using the corresponding buttons at the left bottom corner of the screen (Fig. 5c). Eventually,the CRUD Console retains the capabilities of browsing and exporting data as is the case with the Reader Console.

Note that the above consoles represent a subset of the operations that are supported by our web interface. Apart from them, users have also access to (screenshots are omitted):

– **Online User Registration.** All functionality involving online processing of user requests to get registered in the system and gain (initially Reader's) access to the available data.

– **Batch Upload.** Web interfaces for batch inserting medical devices inventory information and adverse event reports, available only to administrative users.

– **Ad-hoc Triggering of the Matching Engine.** Apart from periodically executing our matching algorithm (Section IV) via scheduled system jobs, administrative users have the ability to access the Matching Engine via the web interface, adjust algorithmic parameters and manually trigger its operation.

– **Email Notifications** composed of potentially hazardous devices to stakeholders. The system periodically checks whether the results of matching have been updated and reviewed (i.e., marked with a "Yes" in the Reviewer Console Fig. 5b) and constructs reports including medical devices along with adverse event reports, which are sent via email to appropriate users based on their role(s). This functionality is especially useful to raise awareness of users neglecting to periodically log in the web interface to consult confirmed matching results.

VI. Conclusions

In this paper we presented the core components of our MEDEVIPAS prototype system for medical devices vigilance and patient safety. Our system matches data regarding reports of adverse events against medical devices inserted by the healthcare providers. We presented an entity matching algorithm that also provides provisions for alternative namings of entities (i.e., manufacturers) that can also assist the vigilance process in multilingual systems, as well as briefly described the capabilities of our web interface.

Acknowledgment

This research has been co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: Thalis. Investing in knowledge society through the European Social Fund.

References

[1] "FDA: U.S. Food and Drug Administration Enforcement Reports," http://www.fda.gov/Safety/Recalls/EnforcementReports/.

[2] "ECRI Institute Medical Device Safety," https://www.ecri.org/Products/PatientSafetyQualityRiskManagement/Pages/ MedicalDeviceSafety.aspx.

[3] "TGA: Australian Government Department of Health, Therapeutic Goods Administration, Database of Adverse Event Notifications," http://www.tga.gov.au/safety/daen.htm.

[4] "IMB: Irish Medicines Board, Reporting Safety and Quality Concerns," http://www.imb.ie/EN/Patients–Public/Reporting-Safety-and-Quality-Concerns.aspx.

[5] "Swissmedic: Swiss Agency for Therapeutic Products, Recalls and other field safety corrective actions," https://www.swissmedic.ch/rueckrufe_medizinprodukte/index.html.

[6] "EUDAMED Vigilance Reports," http://ec.europa.eu/consumers/sectors/medicaldevices/documents/vigilancereports/index_en.htm.

[7] "MHRA: Medicines and Healthcare Products Regulatory Agency," http://www.mhra.gov.uk/#page=DynamicListDevices.

[8] "MEDSAFE: New Zealand Medicines and Medical Devices Safety Authority, Medical Devices, Adverse Event Reporting," http://www.medsafe.govt.nz/regulatory/devicesnew/ 9adverseevent.asp.

[9] M. Samore, R. Evans, A. Lassen, and et al, "Surveillance of Medical DeviceRelated Hazards and Adverse Events in Hospitalized Patients," *JAMA*, vol. 291, no. 3, pp. 325–334, 2004.

[10] P. Andersen, "A suggestion for guidance to the Medical Devices Community on the use of software validation and approval," in *Medical Devices Software Workshop*, 2003.

[11] Z. Bliznakov, G. Pappous, K. Bliznakova, and N. Pallikarakis, "Integrated software system for improving medical equipment management," *Biomed Instrum Technol*, vol. 37, pp. 25–33, 2003.

[12] G. Doukidis, N. Pallikarakis, G. Pangalos, G. Vassilacopoulos, and K. Pramataris, "Edi system definition for a european medical device vigilance system," *Med Inform (Lond)*, vol. 21, no. 3, pp. 233–44, 1996.

[13] M. A. Jaro, "Probabilistic linkage of large public health data files," *Statistics in Medicine*, vol. 14, no. 5-7, pp. 491–498, 1995.

[14] W. E. Winkler, "String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage," in *Proceedings of the Section on Survey Research*, 1990, pp. 354–359.