Data Integrity Module for Data Quality Assurance Within an e-Health System in Sub-Saharan Africa

Jonathan Monda, M.B.Ch.B.,[1] Jeremy Keipeer,[2] and Martin C. Were, M.D., M.S.[3]

[1]Moi University School of Medicine, Eldoret, Kenya.
[2]Regenstrief Institute, Inc., Indianapolis, Indiana.
[3]Indiana University School of Medicine, General Internal Medicine and Geriatrics, Indianapolis, Indiana.

Abstract

Objective: *Ensuring good data quality within telemedicine and e-health systems in developing countries is resource intensive. We set out to evaluate an approach where in-built functionality within an electronic record system could identify data quality and integrity problems with little human input.* ***Materials and Methods:*** *We developed a robust data integrity module to identify, enumerate, and facilitate correction of errors within an e-health system that is in wide use in sub-Saharan Africa.* ***Results:*** *The data integrity module was successfully implemented within an electronic medical record system in Western Kenya. Queries were set to fail if one of more records did not meet defined criteria for data integrity. Only one of 14 data integrity checks implemented uncovered no errors. The other queries had errors or questionable results ranging from 51 records to 30,301 records. However, as a proportion of all patients and all observation, the identified records with likely data integrity problems only constituted a small percentage of all records (mean 0.96%, range 0–4.1%). Twelve of the 14 queries (86%) were executed in<15 s, with the longest query lasting 2 min and 18 s.* ***Conclusion:*** *A tool that allows for automatic data integrity and quality checks was successfully implemented within an e-health system in sub-Saharan Africa. The tool potentially reduces the burden of maintaining data quality by limiting the scale of manual reviews needed to identify electronic records with errors.*

Key words: *e-health, medical records, information management*

Introduction

The past few years has seen increasing adoption of e-health and telemedicine systems in developing countries.[1,2] These systems hold the promise of improved delivery of care services to individuals. They also allow for easier monitoring and evaluation of care, and can potentially improve efficiency of services offered. Central to the success of the implemented systems is the ability to ensure that stored patient data are of high quality. Ensuring good data quality, however, remains a challenge. Forster et al.[3] concluded that the quality of data collected by sites using electronic databases was unsatisfactory for many sites involved in the scale-up of ART in resource-limited settings. Aiga et al.[4] also noted how increasing information requirements by global partners can increase workload of health professionals and subsequently compromise data quality.

It is imperative that approaches are adopted to improve the quality and accuracy of the data collected and stored as a part of e-health and telemedicine system implementations. Currently, most systems ensure data quality by using validation at the time of data collection and entry. This validation is typically done by controlling the type of data that can be entered using choice lists, and by restricting ranges, responses, or field types that are accepted. Although helpful, prevalidation of data still remains insufficient at providing a comprehensive solution to the data quality problem.

Several reasons explain why data quality problems persist even with prevalidation. First, in many e-health implementations in the developing world, data are initially collected on a paper-based encounter form by clinicians and then later entered into the electronic system by data-entry clerks.[5,6] In such cases, busy clinicians might inadvertently miss or erroneously record key pieces of information on the paper forms. Although validation at the time of data entry could catch some of these problems, systems have to be designed to allow data entry to continue, as it is impractical to expect data-entry clerks to stop and contact clinicians every time they find an error. This would be inefficient, expensive, and a big disruption of the clinical workflow. Second, it sometimes takes integration of several pieces of information over time to identify quality problems. As an example, comparing data entered from a verbal autopsy form and those entered for a patient's clinic visit can reveal that the date recorded for the clinic visit is after the patient's official date of death. This type of error cannot be uncovered by simply prevalidating data for individual forms.

A comprehensive data quality solution requires rigorous quality checks not only before and during data entry, but also after the data have been entered into the electronic system. Typically, quality checks after data entry occur through manual human reviews of the entered data. Studies show that the quality of data in a system is directly proportional to the length of time clerks spend on the database.[3] Unfortunately, with human resource and financial constraints in developing countries, clinical sites are typically unable to dedicate many staff hours to audit the large amounts of data that exist in their e-health systems. As a result, data-cleaning efforts happen in an *ad* hoc manner, either in response to errors identified during reporting or to errors identified at the time of a patient's return clinic visit. Such efforts at data cleaning are cumbersome, time consuming, nonscalable, and often miss important errors.

We hypothesized that it was possible to implement a robust data integrity module within an e-health system to replace the human effort needed for postvalidation. This module would identify, enumerate, and facilitate correction of errors in data entered in the system that had escaped initial human and system checks. We describe in this article the development of this data integrity module and its implementation within a large electronic health system used to support HIV care in Western Kenya.

Materials and Methods

SETTING

This work was conducted at clinics affiliated with the partnership between United States Agency for International Development (USAID) and the Academic Model Providing Access to Healthcare (AMPATH) in Western Kenya.[7] This program provides comprehensive care to more than 120,000 active patients who are HIV positive through 27 parent and 31 satellite clinics (*Fig. 1*). Almost 90% of patient visits at these clinics are handled by nurses and clinical officers (equivalent to physician assistants) without the presence of a supervising physician.

E-HEALTH SYSTEM–AMPATH MEDICAL RECORD SYSTEM

USAID-AMPATH clinics use the AMPATH Medical Record System (AMRS) to store comprehensive longitudinal electronic records for all its patients.[8] AMRS was the first implementation of OpenMRS®, an open-source electronic health record system that has subsequently been widely deployed in the developing world.[5] Clinicians at AMPATH do not enter data directly into AMRS but instead fill out paper encounter forms that contain clinical parameters and categorical observations previously defined and coded in the AMRS concept dictionary.[9] Data-entry clerks with basic computer skills and minimal medical knowledge then manually enter the information from these encounter forms into the AMRS.

Patient records in AMRS currently include demographic information, historical and physical examination data, problem lists, and visit data. Laboratory test results stored in AMRS are imported directly from an external laboratory information system using HL7. Unfortunately, AMRS currently lacks medication dispensing information, as it does not communicate with the external pharmacy database. Prescription information from the encounter forms is used to reflect what medications the patient is on.

DATA INTEGRITY MODULE

OpenMRS® functionality can be extended through programming of modules without the need to modify the core system. The Data Integrity Module is one such plugin, and is simply an extension of core OpenMRS® functionality. This module was programmed by a core developer for AMRS with input from clinicians, investigators, and the data management team at AMPATH.

The core part of the Data Integrity module allows users to define the queries to be used for data quality audits (*Fig. 2*). After entering a self-defined query name, users can then select the programming method to be used in defining the query—this ranges from SQL statements, Java- or Groovy-based queries, to using the inbuilt OpenMRS® Logic Service that accepts Arden Syntax. Users can then write in the code for the query, and define how the output should be displayed. Output can be a simple count of number of records not meeting the defined criteria, or a Boolean result. The threshold for logging results can also be defined, recognizing that users occasionally only want to see results after a certain threshold is met. For example, a user can choose to only be informed if more than 30 records have a particular data quality problem, but if the number is less than this, then the query will simply be reported as a 'pass'.

In addition to logging results of the query, the Data Integrity Module allows the user to define several other things, namely (a) additional information to include in the results report; (b) messages to be displayed based on whether the query returned any results; and (c) links to get to each record with an identified error. All user-defined queries can be easily accessed from a single screen, from which they can then be edited, copied, or deleted. In addition, defined queries can be imported into or exported out of the system as XML files with the click of a button.

Once defined, data integrity queries can be executed individually or as a group. Output is stored in a table with a user-friendly interface made available to outline the number and types of errors per query (*Fig. 3*). Users can view the results of the output, or go directly to the

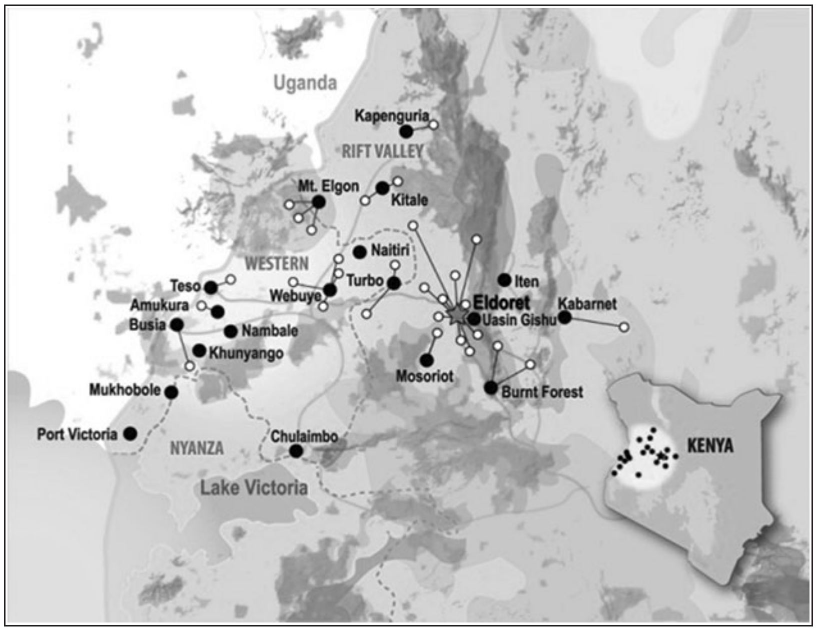
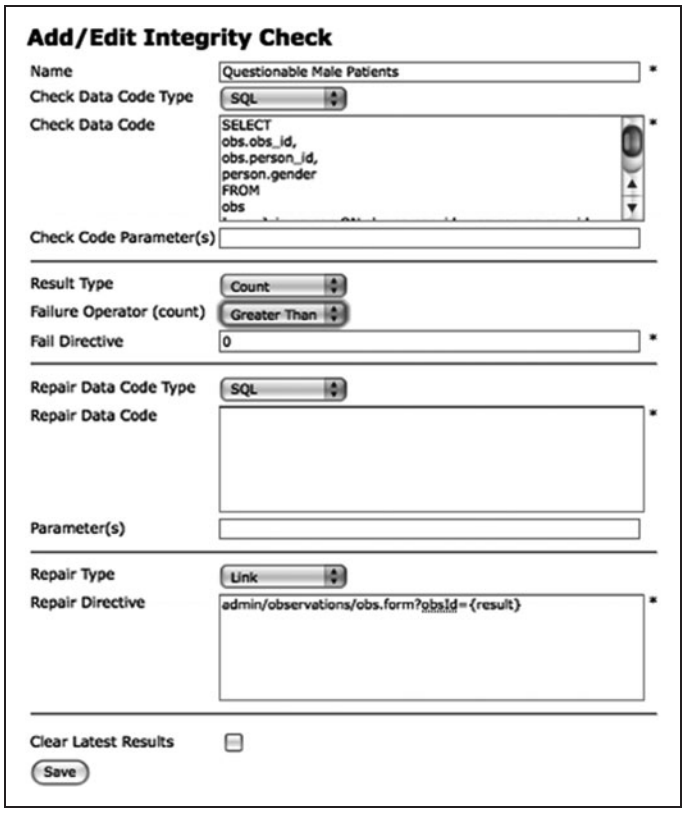


Fig. 1. United States Agency for International Development (USAID)-Academic Model Providing Access to Healthcare (AMPATH) clinical sites.



Add/Edit Integrity Check

Name	Questionable Male Patients *
Check Data Code Type	SQL
Check Data Code	SELECT obs.obs_id, obs.person_id, person.gender FROM obs *
Check Code Parameter(s)	
Result Type	Count
Failure Operator (count)	Greater Than
Fail Directive	0 *
Repair Data Code Type	SQL
Repair Data Code	*
Parameter(s)	
Repair Type	Link
Repair Directive	admin/observations/obs.form?obsId={result} *
Clear Latest Results	☐

Save

Fig. 2. Screenshot for adding and editing a data integrity query.

encounter or observation of interest to do any needed data cleaning (*Appendix Fig. A1*).

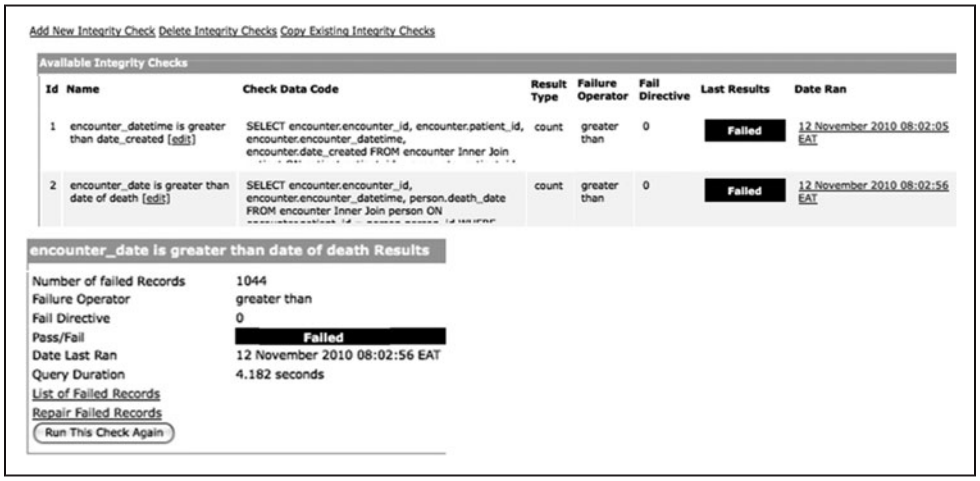
IMPLEMENTATION

The data integrity module to was implemented to identify data quality problems within the e-health system implemented for the AMPATH program. After installing the module, several queries were defined (*Table 1*). The queries included ones that identified wrong types of encounter forms filled for particular patients, erroneous encounter dates, incorrect observations based on gender (e.g., pregnant male), out-of-range observations, and erroneously entered medications. The number of records that failed the data integrity checks and the speed with which the queries ran are reported.

Results

The data integrity module was successfully implemented within the AMRS electronic health system in October 2010. All queries were set to fail if one or more records did not meet the criteria for data integrity. Given the likely length of some queries, the module was run against the research database to avoid interrupting the work in the production system—this research database mirrored all data contained in the production database. Whenever a data manager wanted to find out the records that did not meet the particular predefined criteria for data integrity, they simply ran the relevant query to retrieve results. This was in place of the previous method of manually reviewing thousands of records to identify these quality deficiencies.

Table 2 shows the results from the data integrity checks run at the implementation institution. Of the 14 queries run, only one (which queried for Questionable Data of Death in a patient's record) revealed no errors. The other queries had errors or questionable results ranging from 51 records to 30,301 records. However, as a proportion of all patients and all observations, the identified records with likely data integrity problems only constituted a small percentage (mean 0.96%, range 0–4.1%). Most queries ran quickly, with 12 of 14 queries taking < 15 s to run. The longest query took 2 min 18 s to run, with the length of time for each query directly proportional to the query's complexity and to the number of results returned. For each record that failed the query criteria, a link was created which opened the right record in the production database that allowed users with the appropriate privileges to make needed corrections.



Add New Integrity Check Delete Integrity Checks Copy Existing Integrity Checks

Available Integrity Checks

Id	Name	Check Data Code	Result Type	Failure Operator	Fail Directive	Last Results	Date Ran
1	encounter_datetime is greater than date_created [edit]	SELECT encounter.encounter_id, encounter.patient_id, encounter.encounter_datetime, encounter.date_created FROM encounter Inner Join	count	greater than	0	Failed	12 November 2010 08:02:05 EAT
2	encounter_date is greater than date of death [edit]	SELECT encounter.encounter_id, encounter.encounter_datetime, person.death_date FROM encounter Inner Join person ON	count	greater than	0	Failed	12 November 2010 08:02:56 EAT

encounter_date is greater than date of death Results

Number of failed Records	1044
Failure Operator	greater than
Fail Directive	0
Pass/Fail	Failed
Date Last Ran	12 November 2010 08:02:56 EAT
Query Duration	4.182 seconds

List of Failed Records
Repair Failed Records
Run This Check Again

Fig. 3. Screenshots of data integrity query output.

Discussion

A tool within a robust e-health system in a sub-Saharan African setting was successfully implemented that audited patient records to identify quality and data integrity problems. This tool uncovered thousands of records with errors and provided direct links to these records to allow for easy correction of identified mistakes. Typically, identifying data quality problems would have required many hours of manual human reviews, but our tool was able to perform these functions much faster, and in a reliable and comprehensive way. The tool did not interfere with the function of the

Table 1. Data Integrity Queries

QUERY NAME	QUERY DEFINITION
DOB is NULL or blank	Patients with no identified date or estimated birthdate.
Encounter_datetime is greater than date_created	Abnormal date for a filled encounter form.
Encounter_date is greater than date of death	Records with a clinical encounter date that occur after the official date of death.
DOB is greater than encounter_date	Records with clinical encounter date that occur before a patient's date of birth.
Age < 10 when having adult encounter type	Children who have an adult encounter type entered.
Age > 18 when having peds encounter type	Adults who have a pediatric encounter type entered.
Questionable male patient	Observations in a male patient that would typically be associated with a female patient (e.g., pregnant).
Questionable female patient	Observations in a female patient that would typically be associated with a male patient (e.g., number of wives).
Questionable weight (age < 10 and weight > 40 kg)	Weights that are too heavy to be for a child.
Questionable weight (age > 18 and weight < 40 kg)	Weights that are too low to be for an adult.
Questionable DOB	Erroneously entered date of birth.
Questionable date of death	Erroneously entered date of death.
Patient prophylaxis reported with plan = 'none'	Patient on PCP prophylaxis medications where the plan of care for this medication is 'none'.
Patient no prophylaxis reported with plan = 'continue regimen'	Patient on no PCP prophylaxis where the plan of care is to 'continue regimen'.

DOB, date of birth.

Table 2. Data Integrity Query Results

QUERY NAME	DATE RUN	NO. OF FAILED RECORDS	QUERY DURATION
Encounter_datetime is greater than date_created	Nov-12-2010	1,117 (0.04%)	1.0 s
Encounter_date is greater than date of death	Nov-12-2010	1,044 (0.04%)	4.2 s
DOB is NULL or blank	Nov-16-2010	1,140 (0.4%)	0.0 s
DOB is greater than encounter_date	Nov-19-2010	492 (0.02%)	9.3 s
Age < 10 when having adult encounter type	Nov-12-2010	2,328 (0.1%)	3.1 s
Age > 18 when having peds encounter type	Nov-12-2010	571 (0.1%)	1.3 s
Questionable male patient	Nov-16-2010	11,894 (3.6%)	2 min, 18.7 s
Questionable female observation	Nov-12-2010	112 (< 0.001%)	0.1 s
Questionable weight (age < 10 and weight > 40 kg)	Nov-12-2010	5,106 (0.3%)	13.1 s
Questionable weight (age > 18 and weight < 40 kg)	Nov-12-2010	30,301 (1.5%)	47.8 s
Questionable DOB	Nov-19-2010	51 (0.01%)	0.2 s
Questionable date of death	Nov-19-2010	0 (0%)	0.2 s
Patient prophylaxis reported with plan = 'none'	Nov-20-2010	1,457 (4.1%)	10.5 s
Patient no prophylaxis reported with plan = 'continue regimen'	Nov-20-2010	1,336 (3.2%)	10.7 s

clinics, and the number and types of errors checked could be easily scaled up.

The types of errors uncovered by our tool would be found in any telemedicine and e-health system. These errors occur despite good prevalidation efforts, given that human beings are by nature imperfect.[10] With poor quality of data in repositories, patient care, reporting, and clinical research can be adversely affected. To reduce such errors, it would be ideal if telemedicine and e-health systems incorporated functionality similar to the one described that automatically audited the data entered into the repositories. As observed, the proportion of errors identified constitute a small part of all records and observations, and would be too onerous for individuals to go through these in a manual fashion. A tool similar to the one described would not only save time and resources, but would also potentially improve the quality of care offered to patients.

Several limitations in this work deserve mention. Our tool is custom-built to work within one system, OpenMRS®, and cannot be easily integrated to other systems—however, the idea of having automatic data integrity checks can be applied to any e-health system. This was an observational study, and we did not rigorously evaluate the adequacy of our module at identifying integrity problems against human reviewers. Currently, it still requires some programming expertise (Standard Query Language) to define the queries, and this makes it difficult for lay users to create queries without some form of basic programming assistance.

As a next step, we intend to evaluate the impact of using the data integrity module on several things, namely (a) on the quality of care offered through computerized decision support systems; (b) on time use by data management and data cleaning personnel; and (c) on cost of maintaining data quality in a robust patient-centric data repository. We will also improve the user interface of the module to allow lay users with no programming experience to create the queries they need. Finally, we intend to extend the functionality of the module to allow for aggregate analysis of the returned output—as an example, we hope to be able to create data integrity reports by providers or clinics so as to identify potential targets of interventions to improve data quality.

Conclusion

A tool that allows for automatic data integrity and quality checks was built and successfully implemented within an e-health system in the resource-limited setting of sub-Saharan Africa. Using this tool, thousands of records with data quality and integrity problems were uncovered, thus allowing for easy rectification of mistakes identified. This tool potentially reduces the burden of maintaining data quality by limiting the scale of manual reviews needed to identify records with errors.

Acknowledgments

This work was supported in part by a grant to the USAID-AMPATH Partnership from the USAID as a part of the President's Emergency Plan for AIDS Relief. The authors would like to thank Nyoman Ribeka, Ada Yeung, and staff at the USAID-AMPATH data management for their valuable input into this work.

Disclosure Statement

No competing financial interests exist.

REFERENCES

1. Williams F, Boren SA, Williams F, Boren SA. The role of the electronic medical record (EMR) in care delivery development in developing countries: A systematic review. *Inform Prim Care* **2008;**16:139–145.
2. Mars M. Health capacity development through telemedicine in Africa. *Yearb Med Inform* **2010;**87–93.
3. Forster M, Bailey C, Brinkhof MW, et al. Electronic medical record systems, data quality and loss to follow-up: Survey of antiretroviral therapy programmes in resource-limited settings. *Bull World Health Organ* **2008;**86:939–947.
4. Aiga H, Kuroiwa C, Takizawa I, Yamagata R. The reality of health information systems: Challenges for standardization. *Biosci Trends* **2008;**2:5–9.
5. Mamlin BW, Biondich PG, Wolfe BA, et al. Cooking up an open source EMR for developing countries: OpenMRS—a recipe for successful collaboration. *AMIA Annu Symp Proc* **2006;**529–533.
6. Were MC, Shen C, Bwana M, et al. Creation and evaluation of EMR-based paper clinical summaries to support HIV-care in Uganda, Africa. *Int J Med Inform* **2010;**79:90–96.
7. Einterz RM, Kimaiyo S, Mengech HN, et al. Responding to the HIV pandemic: The power of an academic medical partnership. *Acad Med* **2007;**82:812–818.
8. Tierney WM, Rotich JK, Hannan TJ, et al. The AMPATH medical record system: Creating, implementing, and sustaining an electronic medical record system to support HIV/AIDS care in western Kenya. *Stud Health Technol Inform* **2007;**129(Pt 1):372–376.
9. Were MC, Mamlin BW, Tierney WM, Wolfe B, Biondich PG. Concept dictionary creation and maintenance under resource constraints: Lessons from the AMPATH Medical Record System. *AMIA Annu Symp Proc* **2007;**791–795.
10. McDonald CJ. Protocol-based computer reminders, the quality of care and the non-perfectability of man. *N Engl J Med* **1976;**295:1351–1355.

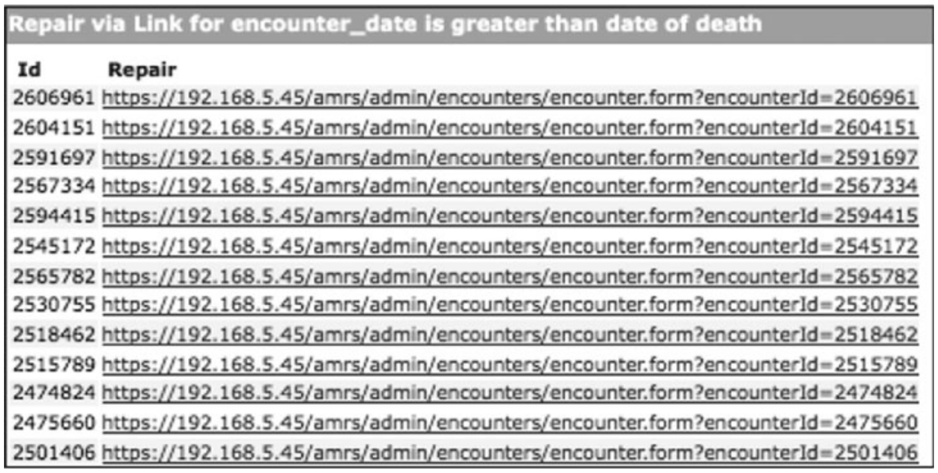
Address correspondence to:
Martin C. Were, M.D., M.S.
Indiana University School of Medicine
General Internal Medicine and Geriatrics
410 West 10th St., Suite 2000
Indianapolis, IN 46202

E-mail: mwere@regenstrief.org

Received: December 20, 2010
Revised: March 3, 2011
Accepted: March 4, 2011

(*Appendix follows →*)

Appendix



Repair via Link for encounter_date is greater than date of death

Id	Repair
2606961	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2606961
2604151	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2604151
2591697	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2591697
2567334	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2567334
2594415	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2594415
2545172	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2545172
2565782	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2565782
2530755	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2530755
2518462	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2518462
2515789	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2515789
2474824	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2474824
2475660	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2475660
2501406	https://192.168.5.45/amrs/admin/encounters/encounter.form?encounterId=2501406

Appendix Fig. A1. Links to encounter forms for records failing integrity checks.