Paper 187-30

PreMIS: Pre-hospital Medical Information System

Jeff Wright, ThotWave Technologies LLC, Cary, NC
Patrick Alston, University of North Carolina, Chapel Hill, NC
Sharon Schiro, University of North Carolina, Chapel Hill, NC
Tim Walters, ThotWave Technologies LLC, Cary, NC

ABSTRACT

PreMIS is a pre-hospital medical records system funded by the State of North Carolina and developed by the Department of Emergency Medicine at the University of North Carolina at Chapel Hill. The system uses web data entry, mobile hand held devices, and batch import to collect incident records from EMS providers throughout the state of North Carolina. This data is then used for bio-terrorism surveillance, provider reports, and billing support. The SAS® System is used by PreMIS to populate a data warehouse and to implement reports. PreMIS reports are surfaced over the Internet through a secure reporting portal via a combination of stored processes, SAS Integration Technologies, and Java. Part of the PreMIS mission is to promote best practices among EMS providers, so great attention has been paid to which performance measures are reported and in what form. This paper will present a history of the implementation to date, and share both successes and lessons learned.

PROJECT INCEPTION

The National Highway Transportation Safety Association's *EMS Agenda for the Future* was published in 1996. The *EMS Agenda for the Future* addresses the need for Emergency Medical Services (EMS) to be fully integrated into the overall health care system. In an effort to meet these goals, North Carolina has developed an electronic EMS data system that enables a standard method of documenting patient care, the *Pre-hospital Medical Information System* (PreMIS). PreMIS facilitates patient care and EMS management practice comparisons across EMS agencies, involvement in public health and injury prevention initiatives, and EMS research.

The project began in the late 1990's with a grant from the Department of Transportation and the Governor's Highway Safety Initiative to the North Carolina Office of Emergency Medical Services. The project was subcontracted to the Department of Emergency Medicine at the University of North Carolina-Chapel Hill with Greg Mears, MD, serving as the Principal Investigator.

North Carolina's Office of Emergency Medical Services (NCOEMS) is within the Division of Facility Services of the North Carolina Department of Health and Human Services. NCOEMS provides several regulatory functions, including ambulance inspections, permission for the operation of ambulances, certification of EMS personnel, and the licensure of EMS providers.

Electronic submission of data to the PreMIS database by EMS systems is mandated according to Title 10, Chapter 3, and Subchapter 3D, of the New North Carolina EMS Rules as approved by Medical Care Commission in 2002 (Effective January 1, 2005):

> Section .2600… (7) County government shall establish EMS Systems. Each EMS system shall have a system to collect data that uses the basic data set and data dictionary as specified in "North Carolina College of Emergency Physician: Standards for Medical Oversight and Data collection";
>
> Section .2602… (3) To receive a designation from the OEMS as a Model EMS System…an EMS system shall document that, in addition to the system requirements in Rule .2601 of this section…a mechanism to collect and electronically submit to the OEMS data corresponding to the advanced dataset and data dictionary as specified in "North Carolina College of Emergency Physician: Standards for Medical Oversight and Data collection";

Before the deadline, many agencies had already begun submit data to PreMIS in order to supplement their data recording methods and quality management capabilities.

The PreMIS project is currently funded through contracts with the North Carolina Department of Public Health and the North Carolina Office of Emergency Medical Services as a fundamental portion of the state's Bio-terrorism Preparedness strategy as supplemented by the Health Resources and Services Administration (HRSA), U.S. Department of Health and Human Services.

SYSTEM OVERVIEW

DEVELOPMENT

The project team began development by defining the mechanisms for entering data and defining data elements to be contained in the PreMIS database. The two initial mechanisms were defined as paper and web-based data entry. The web-based data entry application was developed for Internet access to centralized data storage. This version allows any EMS provider to enter data using a web browser (Internet Explorer). An XML Import method to support third-party applications was developed in November, 2002. Also in 2002, a Palm based mobile application began development to provide EMS agencies with an offline method of data collection.

Several factors were considered in the design of PreMIS including: flow of patient care, ease of data entry, error checking, and intuitiveness. Data points were drawn from several national standards (NHTSA, DEEDS, Utstein, and ICD-9). The database design also includes the ability to link to other data systems including the North Carolina Trauma Registry (NCTR), the State Regional Advisory Council (RAC) database, the North Carolina Emergency Department Database (NCEDD), and the Information System for the State Trauma Advisory Committee (ISSAC).

HARDWARE AND SOFTWARE CONFIGURATION

The servers that house the PreMIS Database, mobile and web applications, FTP (File Transfer Protocol) site, web site, and reporting are all maintained at a commercial hosting facility to provide the greatest degree of security and availability. These servers run on both Windows 2000 and Red Hat Linux Platforms.

Figure 1 is a high-level logical architecture for PreMIS. The data collection web application runs on IBM WebSphere, and stores data in an IBM DB2 transaction database. SAS is used for the Extract, Transform, and Load (ETL) process that loads an Oracle data warehouse. This ETL process is capable of generating e-mail alerts based on technical problems or based on bio-terrorism reports. A second web application serves as a Reporting Portal for analyzing PreMIS data. The overall Reporting Portal is based on Java, but individual reports are SAS® Stored Processes using Base SAS, SAS/GRAPH®, SAS/STAT®, and SAS/QC®. These sub-systems will be explained in detail in the remainder of this paper.

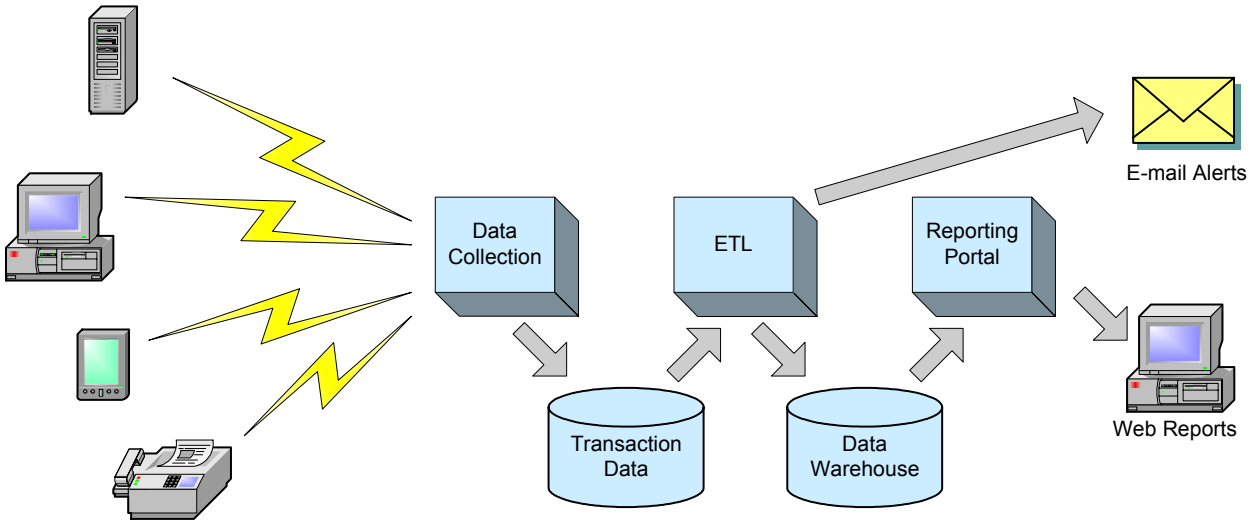

Figure 1

DATA COLLECTION

There are four mechanisms for entering data into the PreMIS database: web-based entry, faxed paper entry, Palm-based device, and import from commercial software.

WEB APPLICATION

The web application method is the base method of data entry. It uses a secure web site for entry into the PreMIS transaction database. This is, as it requires only a computer. This method is available from provider offices, hospitals, or any other place with Internet access and Internet Explorer.

In addition to data entry, this web application provides an Ambulance Care Report (ACR) which documents a single care incident in online or printed form. Java and XML technologies are used to generate this report. The ACR

contains patient and EMS provider demographic information, a description of the patient's illness, procedures and treatments provided to the patient, and disposition information. This report is provided to the hospital to which the patient is taken for care, as this information is important to the patient's emergency room and hospital treatment.

PAPER

Users may currently fax a paper form to be included in the PreMIS database. Paper forms faxed to PreMIS are processed by Cardiff TeleForm, a software package utilizing OCR (Optical Character Recognition). The space constraint of a paper form limits the dataset and therefore the reports available to systems using this method. This method of data entry has been eliminated as of Jan 1st, 2005.

MOBILE

PreMIS Mobile is the Palm-based PDA method of entry into the PreMIS database. For systems that have more personnel in the field than at the station, the goal of PreMIS Mobile was to take into account differing user environments and EMS system workflow. Encrypted data is submitted for inclusion in the PreMIS database via the Internet.

IMPORT

Extensible Markup Language (XML) is used as the method for data to be imported into the PreMIS database. PreMIS XML files must comply with version 1.4, or later, of the PreMIS Input XML Schema Definition. This schema definition file is located at:

http://www.premis.net/Downloads/premis.1.5.xsd.

Data is submitted using a secure FTP client to the FTP Server for processing into the PreMIS Database.

REPORTING

Reporting aspects of PreMIS were jointly developed between UNC and ThotWave Technologies, LLC, a SAS Alliance Partner. The data warehousing strategy focused on providing the necessary data structures, processes, and capabilities in order to generate aggregated information and analysis of emergency medical data. ThotWave's data warehousing methodology was employed on this project to ensure that the PreMIS system was implemented in as short a time as possible with outstanding results. This methodology includes proven industry standard design principles as well as proprietary methodologies honed over many years of building enterprise business intelligence applications.

This section will describe the PreMIS data warehouse, load process, and reports.

DATA WAREHOUSE

Several decision support areas were identified as needing reporting or data extract functionality in PreMIS:

- *Management Reports*: Management reports are directed at operations and performance improvement for EMS Providers.
- *Bio-terrorism Surveillance*: Bio-terrorism surveillance is concerned with monitoring EMS data for patterns of symptom data that could indicate a bio-terrorist attack.
- *Billing Extracts*: Billing extracts are intended to feed an EMS Provider's accounting software with incident records for the purpose of billing.
- *Ambulance Care Report (ACR)*: The ACR is a report of the complete PreMIS record for a single care incident.

In approaching the design of these functions, the starting point was to consider the transaction data model used for PreMIS data collection. This normalized data model requires literally a wall-sized diagram to display its over two hundred data elements and scores of tables. Although it is well-suited for data collection, getting meaningful information out of this raw and fully relational data model required massive SQL joins and intense processing, not to mention an intimate knowledge of the data and business rules. To make the EMS data more usable for decision support applications, it was decided to implement a data warehouse. The plan for the data warehouse was to create a data model that was easier to use for reporting, and to refresh the data from the transaction database on a scheduled basis, initially once per day.

There are two approaches commonly used to create a data model for a data warehouse or data mart. One is to use identified reports to drive the development of the data model. Tables are created that closely match the structure of the data as reported, so that reporting becomes very simple. This approach is quick to implement if report designs are available, but may prove inflexible and costly in storage over time.

The second approach is to actually model the underlying processes that are being represented. For PreMIS, it was decided to create a true model of emergency medical services using a *dimensional data model*. In a dimensional model, the performance measures are stored in fact tables, and contextual attributes are placed in dimension tables. The name *star schema* is also used for this type of design, taken from the pattern that a data model diagram will show each fact table surrounded by the related dimensions, so that the foreign key lines from the fact table to the dimensions form a starburst appearance.

For PreMIS, the most important fact table describes the patient record. The granularity of this table is at the level of one patient and one emergency incident. The dimensions model contextual attributes such as Patient, Scene (where the incident occurred), Symptom, Treatment, Provider, and Destination.

In order to define a manageable scope for the initial design of the warehouse, it was decided to choose a subset of the requirements. After analysis of the ACR use case, it was determined that this needed to be produced from the transaction database, as described above under Data Collection, so this was not part of the data warehouse initial scope. Based on priorities, it was decided that the two areas to drive the initial data warehouse design would be bio-terrorism surveillance and billing extracts. These two areas turned out to be complimentary. Bio-terrorism analysis required a small number of data points, largely associated with symptoms and location. The billing information required a large number of data elements, and required additional patient and insurance details not present in the bio-terrorism requirements.

Note that the ability to do an iterative approach to the data model design was enabled by the initial decision to model the EMS process. If the warehouse design were based on reports, there would have been significant risk that new applications would require significant design effort. However, when the data model reflects the underlying process about which data is being collected, new applications tend to add to the model in predictable and non-disruptive ways.

In developing the dimensional data model for the PreMIS data warehouse, we encountered a modeling challenge that is typical in healthcare applications. One of the characteristics of a dimensional data model is that there is a single dimension value that corresponds to each fact record. In most cases, if you discover in analysis that more than one value of some dimension can be associated with the fact, then it suggests that you have not correctly identified the level of granularity for the fact table. However, there are some legitimate cases for multi-valued dimensions. For example, the patient record fact table in PreMIS is at the granularity of one person and one EMS incident. But there are a number of dimensions, such as symptom, that can have multiple valid values for a single patient record.

We deal with this in the data model by creating a *helper* table to bridge from the fact table to the multiple possible dimension values [KIMBALL02]. Figure 2 shows an example of this kind of structure. Loading this structure involves some fairly intricate transformations, which will be touched on again shortly.

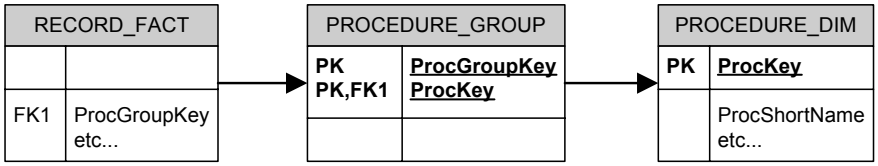

Figure 2

Another situation that can arise in dimensional models is that there can be multiple, related levels of granularity that need representation in the warehouse. When the PreMIS warehouse model was originally created, we believed there were three hierarchically-related fact tables:

- An EMS *Incident*, which has a single Scene and a single Provider, but can have multiple Patients.
- A *Patient Record*, which is a single Patient's data from the Incident.
- A *Vital Signs* entry, which can occur multiple times for a single Patient Record.

In practice, the Vital Signs table has not actually been used as a fact table to date.

The focus for PreMIS data collection is the Patient Record, and this is also the focus of many reports, but the Incident level of detail is also useful. Incidents are not actually modeled in the source database, although there is an incident number that can be used to group Patient Records. In the PreMIS warehouse today, the data model makes an explicit representation of Incidents, but the load process does not do all of the data reduction that could be done to represent this as a different level of granularity. This is an opportunity for further development, because some sorts of analysis (for example, response time) make more sense at the Incident level rather than the Patient Record level of granularity.

The warehouse data model was initially developed through analysis of the source data model and interviews with subject matter experts. The analysis was a process of identifying real world scenarios that can occur in emergency care. In some cases, the scenarios could not be captured correctly because of limitations in the source data itself. This is a common occurrence in data warehousing. The mapping to the source data was created and documented at the same time as the data model. Over time the data model has been expanded as additional applications were built on top of the data.

Oracle was selected as the DBMS for the data warehouse out of cost considerations. After the logical data model was defined down to the necessary level of detail, a physical data model was deployed to Oracle so that the target physical tables were in place, ready for loading.

EXTRACT, TRANSFORM, AND LOAD

Next it was necessary to create an Extract, Transform, and Load (ETL) process to populate the data warehouse. The original approach to ETL, still in use today, was *wipe and load*, meaning that the entire data warehouse would be recreated in each run. A wipe and load ETL process is easy to implement and has some important advantages for on-going support. If there is a failure during the ETL process, the recovery is just to re-run it. Also, if there is an addition to the data warehouse model, no extra data conversion work needs to be done. In other words, you can just deploy the updated ETL process and the warehouse is corrected at the next run. By contrast, with an incremental load approach, an addition to the warehouse model requires not only an ETL update but also a one-time data conversion to load the history of the new addition to the model.

For the wipe and load approach to be used, the data warehouse project needs to meet two criteria: data volume and availability of history. A wipe and load process is clearly less efficient of computer time than an incremental load, so if the data volume is too high the wipe and load process will take too long to run. The requirement for data history is a little more subtle. If the purpose of the data warehouse is to build a historical view of the state of the source data at progressive snapshots in time, wipe and load will not be appropriate. But if the source data contains an adequate history of data, then wipe and load will suffice.

The ETL process was developed using hand-coded logic in Base SAS, version 8.2. Heavy use was made of SAS/ACCESS, PROC SQL, formats, and (of course) the DATA step. The general flow of the ETL process is, for each target data warehouse table:

- Extract a raw or near-raw copy of the source data.
- Transform source data to a structure that matches the target table.
- Truncate the target table and use PROC APPEND to load the staging copy.

The ETL process does not add lineage information to the data, because there is only one source database. However, record source and last updated timestamps present in the source data are brought into the warehouse.

The most common types of transformation performed by the ETL process are:

- **Surrogate key generation** – Artificially generated *surrogate keys* are assigned to dimension records during ETL. At the same time, SAS formats are created to map the dimension natural keys to surrogate keys.
- **Denormalization of dimensions** – Concepts that are considered dimensions in the warehouse are denormalized where practical, to reduce the number of joins required when querying the warehouse. Frequently this involved pulling references from lookup tables into a dimension.
- **Handle optional dimensions** – Some dimension values are optional in the source data. Where this occurs, special "not reported" dimension rows are created so that queries could use a consistent join strategy.
- **Date-time logic** – The source data often stores dates and times separately, complicating analysis of time durations. One important type of report is to measure EMS response times, so some effort is expended in the ETL process to accurately represent date-times in the warehouse. This requires logic to detect incidents that cross over midnight.
- **Helper tables for multi-valued dimensions** – Where there are "group bridge" helper tables, it is necessary to identify the distinct combinations of dimension table entries that are associated with fact records. This is probably the most complex transformation, and involves use of PROC TRANSPOSE.
- **Data reduction** – The source system treats much of its data as new for each patient record. For example, consider symptoms. The source system does have a reference table defining the standard list of symptom codes, but multiple symptoms can be associated with a single patient. There is no logic in the source system to uniquely represent and reference the combinations of symptoms in use. For targeted tables, the ETL process reduces dimension data down to distinct values or combinations of values in use. This makes some dramatic reductions in the size of dimensions that would otherwise have a similar number of rows to the fact table.

There are actually more opportunities to do data reduction than have been attempted to date in the ETL process. One that would have benefits for performance improvement reporting is to build a harmonized representation of incidents from potentially multiple patients involved in the same incident.

The resulting ETL programs are scheduled using ThotWave's SAS Program Agent – Lite edition (SPA-Lite). SPA-Lite is able to read a simple XML file which defines the ETL process as a collection of parallel and sequential steps. Errors cause the sequence to terminate, but can be restarted in mid-sequence after corrective action is taken. SPA-Lite traps all SAS log ERROR and WARNING messages, writing them to a summary log and mailing them to an administrator distribution list. There is also a SAS macro API for defining custom events based on assertions. This system is described in more detail in [NELSON04].

At the time of this writing, the ETL process has the following statistics:

- There are 19 tables in the data warehouse.
- The primary fact table contains approximately 250,000 rows.
- The ETL process takes about 55 minutes and is scheduled once per day.

No effort has been made to performance tune the ETL process to date. Considerable opportunities for parallel processing exist that have not been exploited.

REPORTING PORTAL

The purpose of the PreMIS reporting portal is to give EMS Providers access to data on their incidents in way that facilitates performance improvement, in support of the overall goal of improving emergency medical care in North Carolina. As we set out to build the reporting portal, we had several goals:

- PreMIS reports should present operational data in a way that is useful for operational decision making and statistically valid.
- It should be easy to modify and add reports. PreMIS has several statistical analysts on staff, and it was desired that these people be able to create new reports with a minimum of technical programming effort.
- The portal should be accessible in ways that fit user needs. It should be web accessible and support printing.
- The portal should be secure. Although the current reports do not disclose personal information, it still must be recognized that the system deals with medical records and business confidential information, so security is important.

The PreMIS organization had some SAS expertise in-house, due to their affiliation with the University of North Carolina. The team settled on an architecture where a Java front-end application would provide the reporting front-end, and SAS would be used for individual reports.

The Java application is based on ThotWave's *thinking data*® Toolkit for Business Intelligence. The toolkit provides the navigation model, parameter prompting, and ability to integrate SAS results into a Java web application. When the user logs into the portal, he or she is presented with a menu of available reports by category. After selecting a report, the application prompts for the parameters associated with it. A typical set of parameters would be Provider name and date range over which to analyze incidents. The report can then either be viewed (in HTML format) or downloaded as a print-ready PDF document.

The portal has a number of parameter prompting features that are designed to make users efficient in their use of the portal. Most parameters are prompted for using a picklist that presents only valid choices. These choices are either defined through a database query or a configuration file. Date parameters have sensible defaults, such as first of last month or end of last month, a graphical calendar popup is available for changing date parameters. Parameter values are remembered throughout a session as the user switches from report to report, so that different aspects of the same set of incidents can be viewed in the various portal reports.

Individual reports are created as SAS *stored processes*. Stored processes are parameterized SAS programs that can be called from external languages using the Integrated Object Model (IOM) API, part of SAS® Integration Technologies. Each report stored process consists of two logical modules: a query module, and an analysis module. The *query module* is responsible for efficiently retrieving the relevant data from the data warehouse. This code typically consists of PROC SQL calls using SAS/ACCESS for Oracle, resulting in a WORK data set for analysis. The *analysis module* creates one or more report displays out of the data set, either as tables or charts. Many of these displays use SAS/QC facilities such as PROC SHEWHART.

Many people associate stored processes with SAS9, but PreMIS is based on SAS 8.2, which contains support for an early version of stored processes. Care was taken to emulate certain aspects of SAS9 stored processes, such as the

%stpbegin/%stpend macros and system parameters, so that future migration to SAS9 will be easier. More technical details about both the Java and SAS aspects of how stored processes are called from Java are described in [WRIGHT05].

The report development process begins with a specification created by a subject matter expert. After technical analysis of the report specification, development assignments are made. A technical SAS programmer will be assigned the responsibility for the query module. This person's first task will be to create a *report harness* for the report programmer. The report harness consists of a sample data set and a driver program that emulates the stored process that implements the report. Once this harness is complete, the statistical analyst responsible for report programming can implement the analysis module. The query module is completed in parallel.

When both the query and analysis modules are complete, the report can be deployed and tested. This requires creating a stored process and modifying the reporting portal configuration files. The stored process simply defines the expected parameters and calls the query and analysis modules, typically using %include. The portal configuration files define the portal menu, the library of pre-defined parameters with associated picklist details, and the parameter requirements for each stored process.

There are several points to highlight about this development process:

- No Java programming is required to deploy a new report.
- The most important part of the report, the analysis module, can be effectively developed and tested offline on a PC using the test harness.
- Sometimes the analysis reveals that the report requires data that is not yet present in the data warehouse. In this case a data warehouse and ETL change is also made, but the analysis module can still be developed in parallel.

The portal is secured at several points. First, it is hosted by a secure server (HTTPS), so all web requests and responses are encrypted. This prevents someone from intercepting EMS data or portal passwords. All access to the portal is password protected. Since there was an existing user database in place for the web data entry application, an adapter was created to allow the portal to use this same user database. Finally, a security profile is applied to certain picklists, notably Provider. Users are associated with one or more specific EMS Providers in the user database, and the Provider picklist for reports only shows the authorized list of Providers.

In summary, the technology behind the reporting portal consists of:

- Base SAS, SAS/ACCESS, SAS/Graph, SAS/STAT, and SAS/QC, for creating reports.
- SAS Integration Technologies, for making SAS analytics available as a remote service.
- ThotWave *thinking data* Toolkit for providing a secure reporting front-end.
- Tomcat, an open source J2EE application server.
- The Oracle data warehouse described in a previously.

SAS REPORT DETAILS

The reports provided through the PreMIS Reporting Portal fall into several categories, including: lists of patients, procedures, or treatments; graphical reports that provide information on patient demographics; and reports that allow trending over time.

Documentation of the activities of an EMS provider is provided through several reports. The patient list allows a provider to list all patients seen within a specific period of time. The treatment and procedure lists allow a provider to list the treatments or procedures provided by a specific technician within a specific time period. The procedure list also displays the success or failure of each procedure. Support to providers for billing for EMS care includes reports on loaded mileage and the option of downloading data from the PreMIS database. Loaded mileage is the number of miles driven with patients on board (i.e., from the scene to a patient-care destination). Miles driven to a scene without a patient on board are generally not billable. The loaded mileage reports can be generated either by unit, or aggregated for all units within a provider.

Reports on call volume for specific time periods are important for planning of personnel and equipment requirements. These reports summarize call volume by day of week, time of day, or time of day by day of week. Another group of reports allows the provider to determine the demographics of the population they are serving. These demographics include dispatch complaint, Glasgow Coma Scale (an indication of head injury severity), injury categories, trauma mechanisms (blunt, burn, or penetrating), age, and injury locations (map grids). Pareto charts are graphical reports that show the frequency (or count of events) by category, in descending order of frequency, as well as the cumulative sum of events. Pareto charts are provided for chief complaint, dispatch complaint, dispatch type, and destination name.

Quality management reports provide a mechanism for EMS providers to track the quality of their patient care. For example, a series of reports on the average EMS response time evaluates the time for several response phases:: the time between the 911 call and the notification of the EMS unit, the time between notification of the EMS unit and arrival at the scene, the time between the 911 call and the EMS unit's arrival at the scene, the time between arrival at the scene and departure from the scene, and the time between departure from the scene and arrival at the destination (e.g., hospital or clinic). Several of these response time reports also allow the system to determine the 90% fractile response time. Figure 3 gives an example of such a report; the 90% fractile row is highlighted in color. Reports can be run either for calls that are considered emergent (hot) or non-emergent (cold).

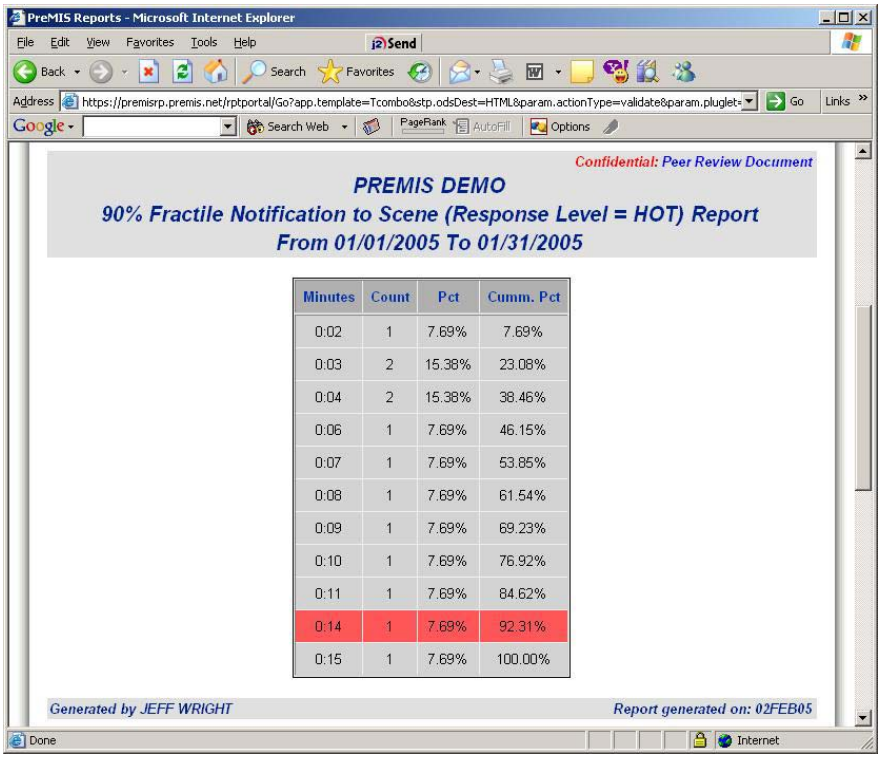

Figure 3

Another form of quality management report allows the users to look at trends in care over time, which allows the provider to look for sudden changes in system performance or deviations in a specific technician's performance as compared to a the average. The reports present the data as control charts, graphing trends over time and delineating average and an acceptable range of deviation from that average. These reports are used to identify trends that may lead to a problem, though no event has yet occurred outside of the acceptable range. This allows providers to identify issues before they become major problems. Figure 4 is an example of such a chart.

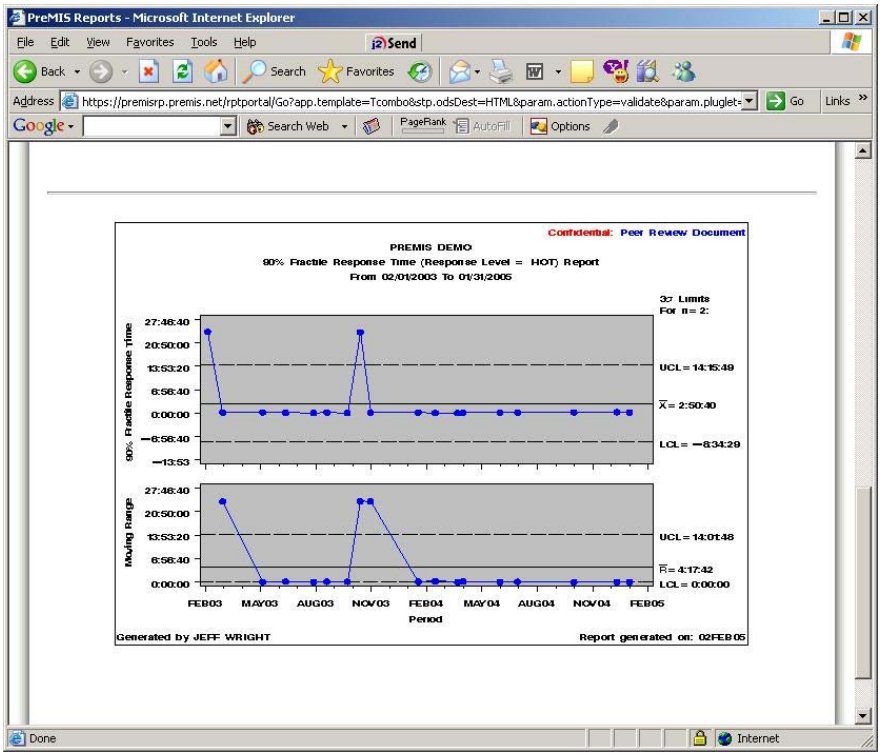

Figure 4

An area of concern in generating these reports is ensuring that the data represented in the report are sufficient to support conclusions that might be drawn. The daily call volume is small for some providers, so control charts and reports based on averages may not contain enough data to be statistically significant. Presentation of data, especially Pareto charts for fields with many values, also has been challenging. Graphical reports with a large number of categories for either the x or y axis tend to wrap, making interpretation of the information difficult.

BIO-TERRORISM SURVEILLANCE

The purpose of the PreMIS bio-terrorism surveillance report is to identify North Carolina counties where a sudden change in pattern of symptoms occurs. The symptoms used are those that have been identified in patients seen by EMS. The report is run on a daily basis, using data from the previous 90 days to evaluate symptom patterns. PROC SHEWHART, specifically the p-chart, is used to evaluate the data.

Symptoms that are evaluated are bleeding, breathing problems, change in responsiveness, choking, diarrhea, drainage or discharge, fever, headache, malaise, mass or lesion, mental status change, mental or psychiatric issues, nausea or vomiting, pain, palpitations, rash or itching, swelling, weakness, and wound. A sudden change in death rate pattern is also flagged.

The output from PROC SHEWHART (Figure 5, Figure 6) allows events to be flagged that are outside control limits (defined as three standard deviations on either side of the average), nine points in a row on one side of the average, six points in a row steadily increasing, or fourteen points in a row alternating up and down.

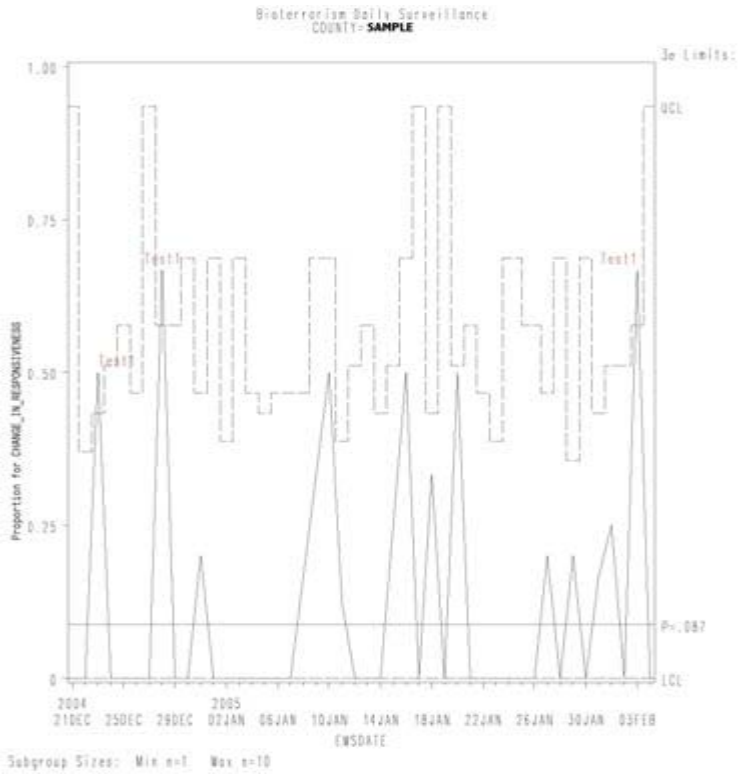

Figure 5

Bioterrorism Daily Surveillance

COUNTY=SAMPL

The SHEWHART Procedure

p Chart Summary for PALPITATIONS

EMSDATE	Subgroup Sample Size	--3 Sigma Limits for Proportion-- Lower Limit	Subgroup Proportion	Upper Limit	Special Tests Signaled
20DEC2004	23	0	0.00000000	0.03346999	2
21DEC2004	25	0	0.00000000	0.03220386	
22DEC2004	28	0	0.00000000	0.03056547	
23DEC2004	23	0	0.00000000	0.03346999	
24DEC2004	16	0	0.00000000	0.03963906	
25DEC2004	15	0	0.00000000	0.04085827	
26DEC2004	11	0	0.00000000	0.04729901	
27DEC2004	29	0	0.00000000	0.03007670	
28DEC2004	24	0	0.00000000	0.03281714	
29DEC2004	20	0	0.00000000	0.03571429	2
30DEC2004	35	0	0.00000000	0.02759862	
31DEC2004	14	0	0.00000000	0.04220588	
01JAN2005	24	0	0.00000000	0.03281714	
02JAN2005	19	0	0.00000000	0.03657810	
03JAN2005	18	0	0.00000000	0.03751293	
04JAN2005	20	0	0.00000000	0.03571429	
05JAN2005	18	0	0.00000000	0.03751293	
06JAN2005	28	0	0.00000000	0.03056547	
07JAN2005	24	0	0.00000000	0.03281714	2
08JAN2005	30	0	0.00000000	0.02961257	
09JAN2005	6	0	0.00000000	0.06317122	
10JAN2005	28	0	0.00000000	0.03056547	
11JAN2005	22	0	0.00000000	0.03416686	
12JAN2005	23	0	0.00000000	0.03346999	
13JAN2005	34	0	0.00000000	0.02796558	
14JAN2005	26	0	0.00000000	0.03162631	
15JAN2005	8	0	0.00000000	0.05503787	
16JAN2005	10	0	0.00000000	0.04948740	2
17JAN2005	24	0	0.04166667	0.03281714	1
18JAN2005	18	0	0.00000000	0.03751293	
19JAN2005	20	0	0.00000000	0.03571429	
20JAN2005	19	0	0.00000000	0.03657810	
21JAN2005	18	0	0.00000000	0.03751293	
22JAN2005	17	0	0.00000000	0.03852908	
23JAN2005	18	0	0.00000000	0.03751293	
24JAN2005	19	0	0.00000000	0.03657810	
25JAN2005	25	0	0.00000000	0.03220386	
26JAN2005	25	0	0.00000000	0.03220386	2
27JAN2005	20	0	0.00000000	0.03571429	
28JAN2005	25	0	0.00000000	0.03220386	
29JAN2005	29	0	0.00000000	0.03007670	
30JAN2005	31	0	0.00000000	0.02917109	
31JAN2005	21	0	0.00000000	0.03491293	
01FEB2005	21	0	0.00000000	0.03491293	

Figure 6

Many of the events that were flagged in the first version of these reports represented clinically insignificant changes in symptom patterns. For example, a provider that rarely sees patients with headaches, then has one patient with a headache, will get flagged as having a potential bio-terrorism event - since the percentage increase is large. This type of flag is seen as a false positive due to the low total patient count. Flagged events where a provider saw zero patients with a specific symptom for nine or more days also had to be eliminated.

At this point in our system development, the bio-terrorism report is initiated manually by running a SAS program on a daily basis. Each county-symptom combination that is flagged as a possible event must be reviewed. Review of these events involves re-running PROC SHEWHART to generate a p-chart for that specific county-symptom combination. This process is very time consuming. Efforts are underway to automate the generation of the graphs of interest, and to provide access to these graphs through the PreMIS reporting portal. This automation will include using the SAS Program Agent described above to send email and/or pager notifications to the person responsible for reviewing the reports, so that the reviewer only need access the report portal if events are flagged.

SUCCESSES AND LESSONS LEARNED

Successes of the project include the following:

- PreMIS is being supplied with data from across the state of North Carolina, and usage is growing.
- A data warehouse design targeted at bio-terrorism surveillance and EMS management reporting was created.
- The dimensional "star schema" data warehouse design based on a model of emergency medical services has proven to be straightforward to extend as additional data elements have been required for reporting.
- The "brute force" wipe and load ETL process has been simple to support and has allowed easy modification to the data warehouse.
- The architecture of the reporting portal has been successful in allowing statistical analysts to create new reports with limited assistance from technical programmers.
- Statistical quality control methods have been useful for identifying statistically significant information for performance improvement.

Lessons learned include:

- Data quality issues require a lot of time to support, particularly when developing organization to organization feeds of information.
- The business users were very busy during development and didn't get an opportunity to check for data quality until after the reports were in beta on the production server, which is not ideal. In general, the team was not as disciplined with data validation and acceptance testing as should have been the case.
- For ETL development, plenty of attention should be given to defining a productive development environment for ETL. It is frequently not possible to do daily development against a production source data system, for a number of reasons. For this project, we started with a snapshot of source data in the form of SAS data sets on the target platform (Linux). Later on in the process some reduced snapshots were created and manipulated on the ETL programmer's Windows PC, which was a more efficient development environment. Some differences between reading from data sets and the source DB2 tables were discovered late in the project.
- On a related note, several times space limitations were encountered on the test server. ETL programming (and testing) can be demanding of system resources.
- Source code management should have been implemented earlier in the project, to save time and reduce risk.

FUTURE DIRECTIONS

Major improvements to PreMIS are underway for 2005, with a particular emphasis on overhauling the data collection architecture. Planned enhancements include:

- Support for import of EMS data using the National EMS Information System XML schema [NEMSIS].
- Further customization of the web application as a data entry and retrieval mechanism.
- Export of PreMIS data for use in provider level applications and analysis.
- Reporting system changes to provide a mechanism for agencies to make operational decisions to improve EMS performance.
- New reports for key clinical measures in EMS.
- More efficient back-end databases and web applications.

REFERENCES

[KIMBALL02] Kimball, Ralph and Margy Ross. "The Data Warehouse Toolkit: the Complete Guide to Dimensional Modeling." Wiley: 2002 (2nd edition).

[NELSON04] Barnes Nelson, G.S and Jeff Wright. "Automated Testing and Real-time Event Management: An Enterprise Notification System ". Presented at the SAS Users Group International Conference, Montréal, Canada. May, 2004.

[NEMSIS] National EMS Information System web site, http://www.nemsis.org.

[PREMIS] North Carolina Pre-hospital Medical Information System web site, http://www.premis.net.

[WRIGHT05] Jeff Wright. "Catch the Stream: SAS® Stored Processes, ODS, and Java." Presented at the SAS Users Group International Conference, Philadelphia, Pennsylvania. April, 2005.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

jwright@thotwave.com
phalston@med.unc.edu
sharon_schiro@med.unc.edu
twalters@thotwave.com