Implementation Brief ■

Integrating Data from Legacy Systems Using Object Linking and Embedding Technology:

Development of a Reporting System for Heavy Metal Poisoning Results

SHANG-CHE LIN, MS, MBA, KAVOUS ROUMINA, PHD, ADAM FADLALLA, PHD, WALTER H. HENRICKS, MD

Abstract Integrating data that reside in different systems remains an often laborious process, requiring either manual steps or complicated programming. This paper describes a method for state-mandated reporting of childhood blood lead testing results that makes use of object linking and embedding technology and readily available software products to pull together information from different legacy systems. A terminal session emulator employs object linking and embedding automation to extract host data, and Visual Basic routines specify the user interface and database manipulation. This system has significantly increased the efficiency and accuracy with which blood lead testing reports are provided to the local state health department. The system provides a model for a relatively easy solution for laboratories and other groups that need a way to integrate standard data sets that are distributed across legacy systems.

■ **J Am Med Inform Assoc.** 2000;7:357–360.

Integrating data from legacy systems remains one of the greatest challenges in health care informatics. Laborious and expensive methods are often necessary to combine information from two or more different systems. Information requirements for reporting childhood blood lead testing are a specific example of such a situation.

Lead poisoning in children remains a major preventable environmental health problem in the United States. Approximately 1 million children in the United States have blood lead levels of at least 10 μg/dL, a level that has been linked with cognition deficits, learning disabilities, and other neurodevelopment defects.[1,2] The Lead Contamination Control Act of 1988 authorized the Centers for Disease Control and Prevention (CDC) to initiate program efforts to eliminate childhood lead poisoning in the United States.[3] The CDC has recently issued revisions to its original 1991 guidelines for childhood lead screening.[4] The guidelines recommend that state health departments require clinical laboratories performing lead tests to report all lead test results and other important demographic information for persons on whom they perform testing. In Ohio, the Department of Health collects all laboratory blood lead, mercury, cadmium, and arsenic results from clinical laboratories performing such tests. The information that the clinical laboratory must report to the department typically resides in different systems, including the laboratory information system and the hospital information system. Performance of such tasks requires significant personnel time and is subject to human error.

In this paper, we present a system that uses object linking and embedding (OLE) technologies and off-the-shelf tools to automate the search for and collection of legacy system data and transform these data into a format necessary for manipulation and placement into a PC-based database, in this case for heavy metal poisoning result reporting to governmental agencies.

Affiliation of the authors: Cleveland Clinic Foundation (S-CL, KR, WHH) and Cleveland State University (AF), Cleveland, Ohio.

Correspondence and reprints: Shang-Che Lin, L22 Laboratory Information Services, Cleveland Clinic Foundation, 9500 Euclid Avenue, Cleveland, OH 44195; e-mail: ⟨lins1@ccf.org⟩.

Received for publication: 12/8/99; accepted for publication: 1/31/00.

Methods

Previously, to fulfill mandates of the Ohio Department of Health, laboratory technicians at The Cleveland Clinic Foundation first accessed the laboratory information system to produce a printout of all the heavy metal results for a given day. Then, each patient's medical record number was entered into the hospital information system to derive patient demographics. These two sets of data were then manually entered into a PC-based database (Paradox) for storage and subsequent submission to the central state health department database. To gather all information from the laboratory and hospital information systems, a technician needed to perform 16 steps in the laboratory system and navigate through six screens in the hospital system for each patient. This entire process required a minimum of fifteen to twenty minutes of technician time per patient. The data required by the state health department included patient, employer, provider, guardian, test result, and laboratory location information, comprising a total of 45 fields.

To increase the efficiency and accuracy of the process of reporting heavy metal testing information, an automated solution was developed. The heavy metal reporting (HMR) system described here uses object linking and embedding (OLE) technologies to automate the various tasks involved. The system is written in Microsoft Visual Basic 6.0[5] and uses Attachmate Extra![6]

Visual Basic serves the purposes of designing the graphical user interface, implementing the application logic, and accessing Microsoft Access databases in the HMR system. The system uses the OLE DB and ActiveX data object data access methodologies provided by Visual Basic. OLE DB, the latest version of Microsoft-supported Open DataBase Connectivity, comprises two components, a data provider and a data consumer. A data provider, Microsoft Access in the HMR system, is an application that responds to queries and returns data in a usable form. A data consumer is an application that uses the interface of OLE DB to access a data source.[7] The HMR system uses a high-level data access model, ADO, as the data consumer to manage and display data-source data.

Attachmate Extra! is an emulation tool for connection between a PC and a network host (i.e., the legacy system). The central structure of Extra! is a session, which encapsulates the connection and application information for interactions with a host system. The Extra! session is a programmable object, whose built-in functionality is accessed by user-written applications. Extra! supports OLE Automation (a Microsoft Windows standard for inter-program communication), which allows users to copy "objects" between applications, with each "object" containing enough information about its format and its creation application to work in a variety of OLE-enabled applications. The crucial functionality here is that data fields in the hospital information system screen can be actively accessed by the OLE automation capabilities of the session emulator, transforming these fields into data that can be inserted into text files and, ultimately, into the database.

The general structure of the HMR system is shown in Figure 1. It is been implemented on an Intel 350 MHz PC with 64MB RAM running Windows 98 in a local area network environment. The file server is a Novell server holding text and database files. The HMR system consists of two Extra! sessions that access the laboratory and hospital information systems, and four Visual Basic modules that manipulate files and control the database. Figure 2 illustrates in a flow diagram how these sessions and modules work together in HMR to produce an integrated output file ready for transfer to the state health department.

Observations

The overall process works as follows. First, the HMR system opens a session with the laboratory information system to collect the previous day's test results and patient medical record numbers from the system. These data are placed into a text file that is transferred to the file server. Next, the HMR system closes the laboratory system session and opens a session with the hospital information system. Using the medical record number obtained from the laboratory system session, the hospital system session captures complete patient demographics from several hospital system functions. These data are also captured into various

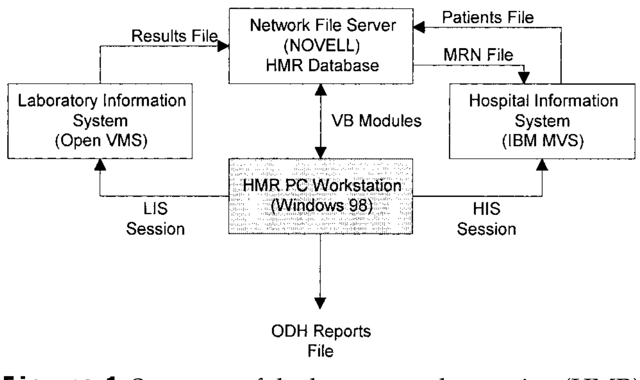
Network File Server (NOVELL) HMR Database
Results File
Patients File
MRN File
Laboratory Information System (Open VMS)
VB Modules
Hospital Information System (IBM MVS)
HMR PC Workstation (Windows 98)
LIS Session
HIS Session
ODH Reports File

Figure 1 Structure of the heavy metal reporting (HMR) system. MRN indicates medical record number; LIS, laboratory information system; HIS, hospital information system; ODH, Ohio Department of Health.

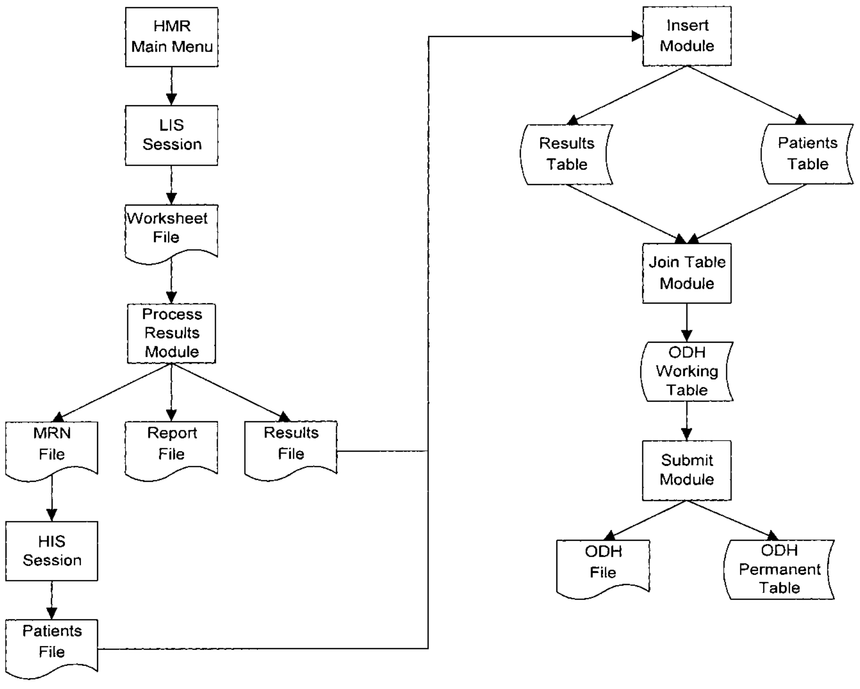

Figure 2 Flow diagram of the heavy metal reporting (HMR) system. LIS indicates laboratory information system; MRN, medical record number; HIS, hospital information system; ODH, Ohio Department of Health.

text files and returned to the file server. A Visual Basic module joins the test results and the patient information into a temporary database table, with patient medical record numbers as the unifying field. At this point, only data not automatically captured from either the laboratory or the hospital information system —specifically, the physician/provider identification number—is manually entered by the laboratory technician. Such data are entered only once and are stored in a separate database table. The HMR system's graphical user interface presents a drop-down list from which the user chooses physician/provider information. Finally, all required information is stored in a permanent Microsoft Access database, formatted, and then submitted in a report to the Ohio Department of Health.

The implementation of the HMR system has significantly reduced the workload of laboratory technicians. The system has saved approximately 50 full-time-employee hours per week in the processing of roughly 120 patient records. As a result of implementing the HMR system, transcription errors have been virtually eliminated, and all reports are provided in a timely manner and in compliance with the department's requirements. A total of 80 programmer hours were spent developing and implementing the HMR system, and the system has required no significant maintenance. The only system failures have occurred as a result of external factors, such as occasional network failures and unavailability of the laboratory or hospital information system.

Discussion

This paper describes an innovative method using OLE technology to integrate data residing in legacy systems. The HMR system is a solution that employs standard, commercially available products to pull together blood lead laboratory results and other information from disparate legacy databases. The key to the system is that the combined OLE automation of the emulation tool (Extra!) and OLE DB of the graphical user interface programming language (Visual Basic) work together to transform static, on-screen data into a dynamic electronic format with minimal manual intervention. The HMR system automates data entry and enables the clinical laboratory performing the tests to comply more efficiently with state health department mandates by providing the data in a timely, accurate, and standard fashion.

One of the most important elements of the HMR system lies in the use of the Extra! tool to read patient medical record numbers from the medical record number file originally derived from the laboratory information system and to enter each number as an object appropriately into the hospital information system in order to query demographic data. This feature automatically links the patient-specific information from the different legacy systems and enables integration into the state health department database.

An alternative method of data exchange, known as Health Level 7 (HL7), was considered and rejected because of cost and lack of flexibility. Our laboratory and hospital information systems are both vendor-supported. To obtain HL7-formatted data from either system, we would have needed to rely on their support. However, the low volume of test results in this project (in comparison with the number of test results provided by the laboratory information system to other interfaced systems in the HL7 format) made the request to the laboratory system vendor cost-prohibitive. In addition, the request for appropriate data format would have delayed the project because of a backlog of requests made to the vendor. The hospital information system was incapable of providing HL7-formatted data.

Moreover, the methodology we have used is flexible and easy to maintain. We can easily and quickly adapt to different end-user requirements without the vender's involvement. Therefore, our model provides a solution for low-volume, special-purpose reporting (e.g., to meet research and regulatory requirements).

A number of important benefits have been realized as a result of implementation of the HMR system. Full-time employee time spent on manual data retrieval and entry has significantly decreased, as has the number of transcription or keystroke errors. The capacity to absorb a higher testing volume has increased as a result. The state health department receives reports in a timely, accurate fashion and, consequently, the laboratory is in better compliance with state mandates and has improved its contribution to public health efforts.

Attractive features of the method employed here include its relatively low cost and ease of development. Standard, off-the-shelf tools are used, and significant but not prohibitive programmer time and knowledge are required. Legacy programming code need not be rewritten, nor is a customized application interface necessary. One minor limitation is the need to modify the programming in the emulation sessions if legacy systems change, but the tools are flexible enough that such modifications can be accomplished readily. The HMR system described here could be a model for a relatively simple solution for laboratories and other entities that need a way to integrate standard data sets that are distributed across different legacy systems.

References ■

1. American Academy of Pediatrics, Committee on Environmental Health. Screening for elevated blood lead levels. Pediatrics. 1998;101(6):1072–8.
2. Jackson RJ, Cummins SK, Tips NM, Rosenblum LS. Preventing childhood lead poisoning: the challenge of change. Am J Prev Med. 1998;14(3S):84–6.
3. Centers for Disease Control and Prevention. Preventing lead poisoning in young children. Atlanta, Ga: CDC, 1991.
4. Centers for Disease Control and Prevention. Screening young children for lead poisoning: guidance for state and local public health officials. Atlanta, Ga: CDC, 1997.
5. Fronckowiak JW, Helda DJ. Visual Basic 6: Database Programming. Foster City, Calif: IDG Books Worldwide, 1999.
6. Attachmate Extra! Personal Client. OLE Automation Programmer's Reference. Bellevue, Wash: Attachmate Corp., 1996.
7. Blakeley JA, Pizzo MJ. Microsoft Universal Data Access Platform. Proc ACM SIGMOD Int Conf Manage Data. 1998:502–3.