Approaches to Integrating Data within Enterprise Healthcare Information Systems
David N. Mohr, M.D., Steven D. Sandberg
Mayo Foundation, Rochester, Minnesota

ABSTRACT
The benefits of an Enterprise Healthcare Information System are related in large part to the degree with which its data and the processes it supports are integrated. There are several technical approaches to achieve integration. The strategic decision to put a group or several groups of applications within a single product to improve integration depends on the degree to which best of breed solutions or an integrated whole is needed. It also depends on a number of factors specific to each organization. It is important to understand the challenge of interfaces before choosing the best solution.

One of the major objectives of healthcare organizations over the next few years is to implement a computer-based medical record (1). Depending on the definition, this may or may not fulfill all the business needs of most organizations or improve quality of healthcare and reduce costs. What most organizations need is an integrated information system that supports the clinical, research, education, workflow, and business processes of healthcare. This system would ideally support care from the initial interaction of the patient with the system to the conclusion of an episode of care. All information would be available for any interaction of the patient with the healthcare system without need for redundant questions or redundant data entry by healthcare providers. For example, source systems would be notified of appropriate scheduling information and the status of an ordered test would be available to all. All information and outcomes would be available for analysis to determine the best approaches to care. This degree of information integration will certainly improve healthcare (2). It is very difficult to achieve, however, and it requires different strategies for different organizations.

Technical Options for Integration
The integrated information system just described can be achieved technically by the healthcare organization in several ways. All require an organized information model with clear data definitions and clear relationships between data elements. The first approach is to use applications developed by a single development group in a unified database with multiple modules to accomplish all the needs of the organization. It is unlikely that any one development group could produce such an application, although key portions of an organization’s needs can be met by this solution as will be discussed below.

An alternative approach for a healthcare organization is to use the “best of breed” application for each module and then interface these applications together to achieve integration. This assumes different development groups from within or outside the healthcare organization produce solutions for the various information needs and that the integration goal be accomplished by an interface engine which ties all the systems together. Because such solution has its own database, unique user interactions, and unique interaction models exist and because the interfacing is difficult, this has not been an acceptable solution for integration (3).

A third approach is to use component based object oriented technology for all modules and achieve the integration goal through this technology and a distributed database (4). This solution relies on standards, future adoption of this technology by solution vendors, and the belief that different builders of these components will have the same data definitions and concepts to allow their components to interact.

Choosing the Best Integration Strategy
Currently most organizations use a combination of the first two solutions described above. There are some application domains in which a system built by a single development group supplies all the applications within it, and some domains in which best of breed systems are interfaced together to achieve some degree of integration. These domains include patient administrative systems, common clinical systems, source or departmental systems, billing systems, financial systems, analysis or decision support systems, and practice support or managed care systems. Between domains the same combination of interfaced and single developer solutions to achieve integration applies. It should be noted, however, that having a single vendor for a domain or group of domains does not always guarantee that the applications are highly integrated.

When there is a need to support common processes, a need for common data definitions, and a need for a common user interface, a single development group should be used to build the approach to a group of applications, the domain of applications, or the cluster of domains of applications. An interfaced, best-of-breed approach should be used when the information flow between applications or domains is minimal. Best of breed solutions allow the simplicity of administratively separate teams, decreased technical uniformity requirements and the ability to purchase off the shelf solutions for a particular need. When there is a need for close integration, however, the use of interfaces begins to consume resources and slow the ability of the enterprise to respond to technical changes. These hidden costs of interfaces occur because processes that are supported by the different systems may be so different, unless developed with integration in mind, there may be no way to map a concept or data element from one application to the other. Even if conceived with integration in mind, data elements in each application may take on new meanings and drift with time as the application responds to changing workflow or business
needs by squeezing new concepts into tables and fields designed earlier.

In addition to the considerations above, organizations have unique features which also drive the approach to an integration strategy. If legacy systems are in place that are supported by a single vendor or developer, integration is easier within that technology and additional functionality in any domain may be more easily achieved with an addition to the legacy solution. Similarly the expertise of the organization will drive what application is chosen, whether or not it is best of breed or part of an integrated package. If the business strategy of an organization is to move to a particular technology or there is an IT strategy towards a particular architecture, that would drive the decision one way or another.

**Example Integration Approach**

Our organization's approach to implementation and integration of its clinical systems can illustrate many of the points above. Our approach can be explained by starting with the common application domains that are in use at most medical centers. These domains as described above include patient administrative systems, common clinical systems, source or departmental systems, billing systems, financial systems, analysis or decision support systems, practice support or managed care systems, and the electronic medical record (EMR) repository.

Since we had finished an extensive many year roll out of a registration system and scheduling system, we were reluctant to intrude on these applications within the patient administrative system domain. The functionality in these areas was complex and the software was tailored to numerous processes that would be difficult to support in another application without significant modification of the processes. Similarly the revenue systems were recently implemented throughout the inpatient and outpatient environment. The managed care system was in place and integrated in some degree to the scheduling system. Because the main need for functionality fell within the clinical systems domain, it was decided to implement this domain with a single integrated product (including orders, results, notes, inbox, medication ordering, nursing documentation, flowcharting, and order communication) and to use the same product for the EMR repository domain. This resulted in a commonly developed solution for the patient care domain (PCD) and the domain of the EMR repository. In addition, because the pharmacy application, which was in the departmental system domain, could not be adequately interfaced to the PCD applications without extremely complex interfaces, it was decided to tightly integrate this application with the solution chosen for the clinical systems product. We chose not to use many of the other applications provided by the vendor available for the other domains of our clinical systems. The benefits of this decision were that we avoided the disruption of changing the existing implementations at the user side for non-clinical system domains, and we kept the software that had been enhanced in these areas over the years.

Once the scope of out patient care domain (PCD) and the EMR repository was defined, the next step was to define the interfaces that must exist at the boundaries of this domain.

The definition of these interfaces and how the data must be translated was a major effort. What follows is a discussion of these interfaces and the effort required dealing with inconsistent definitions of shared data elements.

The effort required to build an interface depends on the different application and data elements that are used in common. There are many data elements that need to flow between applications in the new PCD and applications in other domains. These elements include patient information, provider information, department information, location information, etc. To facilitate these interfaces, much of this information is kept in reference files. A unique code is assigned to each record in this file. It is this code that is passed between applications. As each of these data elements is passed from on application to another it is necessary to ensure that the definition of these data elements is consistent. If the application can share these reference files then the consistency is guaranteed. However, if the applications are developed on different platforms or by different vendors then the consistency of these data elements must be maintained by some other mechanism. While the general concept of these data elements may be similar, the different applications tend to implement them in slightly different ways. These differences also tend to increase as the application evolves and matures over time. The result of this is that when these applications must be integrated a translation table must be created and continually maintained to translate the codes from the source application into the codes used by the target application.

The following example illustrates the effort required to integrate two data elements (department and location) from external applications, through a sampling of applications within the patient care domain (PCD), and finally on to the billing applications. Departments are the organizational entities within the clinic responsible for a set of activities. Locations are the physical places within the clinic at which activities occur, in particular two types of locations were used, the location to which a patient is to report and the locations to which the medical record is sent.

PowerDesign® Process Analyst™ 6.1 was used to create a Data Flow Diagram (DFD) to document the relevant data elements as they flow into and out of the various applications, see diagram. The square boxes represent the external entities, those applications outside the scope of the single integrated product used for the applications in our patient care domain (PCD). The rounded boxes (#3, #4, #13) represent the applications within the PCD. These are the applications that act on the data by some means, either programmatically or on screens. The open-ended rectangles are the data stores, that is, the database files in which the data is stored between applications.

Two scheduling applications were involved, the Central Appointment Desk (CAD), a rules based scheduling application, and the General Purpose Appointment System (GPAS), a resource based scheduling application. Both of these applications create appointments to be used within the PCD. The History Location System (HLS) tracks the medical record to the "home" department for a patient's episode. The Medical
Record Locator (MRL) application then tracks the medical record to other locations based upon the patient’s appointments. The Check-in/Patient Locator application tracks the location of the patient as the patient arrives at each area, this application also creates an Encounter record which documents the interaction between a patient and a provider. The Problem List/Service Recognition (PL/SR) application documents the problems involved in the Encounter and creates a Charge Record for the Encounter. These Charge Records are used by the Medical Revenue System (MRS) to determine the billing amounts for the patient encounter.

All of these applications use the concepts of department and location in some fashion. The following table illustrates how these data elements were translated from code set to another as the applications were integrated. HLS uses a Section code and a Location code to indicate the location of the medical record. These codes had a definition similar to the Department and Chart Location within the PCD. Therefore, a cross-reference table was used in the HLS Interface to translate the code from HLS to the corresponding code within the PCD.

CAD uses Quotas and Procedures to determine who is responsible for an appointment. A Report-To location may be derived from the Quota and Procedure or may be manually entered. Extensive programming logic is required in the CAD Interface to derive a Department based upon the Quota and Procedure. The Report-To location is simply translated into a PCD Report-To Location. The Chart Location is then derived based upon the Report-To Location.

GPAS uses Site, Calendar and Appointment Type to determine who is responsible for an appointment. The Report-To location may be derived based upon the Site, Calendar and/or the Appointment Type, or it may be manually entered. Extensive programming logic is required in the GPAS Interface to derive a Department based upon the Site, Calendar and Appointment Type. The Report-To location is simply translated into a PCD Report-To Location. The Chart Location is then derived based upon the Report-To Location.

MRL uses the Department and Chart Location of the Appointment to track the location of the medical record. Since the Appointments and Medical Record Location data stores are both within the PCD, they both relate to the same reference files for Departments and Locations. Therefore, no translation is required between these datasets.

The Check-In/Patient Locator application uses the Department and Report-To Location of the Appointment to track the location of the patient. Since the Appointments and Patient Location data stores are both within the PCD they both relate to the same reference files for Departments and Locations. Therefore, no translation is required between these datasets. However, the “departments” required for an Encounter are different than the departments for an Appointment, therefore they are referred to as Service Groups. The
Service Groups are at a more detailed level than Departments (i.e., each Department may relate to many Service Groups) and therefore must be manually entered on the screen. This Service Group, along with the Report-To Location from the Appointment, is used to create the Encounter record.

PL/SR documents the Problems associated with an Encounter. PL/SR also documents the charges for the Encounter. The Service Group from the Encounter is used to categorize the charges. Since the Encounter and the Charge Record data stores are both within the PCD, they both relate to the same reference files for Service Group. Therefore, no translation is required between these datasets.

MRS uses Work Area, Service Area, Revenue Number and Cost Center to categorize the charges. Extensive programming logic is required in the MRS Interface to derive these data elements based upon the Service Group. A Service Group will map directly into one Work Area and one Revenue Number. Each Work Area can then be mapped to a Service Area and each Revenue Number can be mapped to a Cost Center.

It is difficult to estimate the exact effort required for resolving the mapping of these data elements. Initial meetings of large groups were required to understand the problem and assign people with the proper expertise to solve the issues. Several people were involved from each of these application areas, both from the technology areas to determine how the translation was to be done and from the business areas to determine what codes were to be mapped. The initial effort took several months. It should be noted that this is an ongoing effort, the mapping of these codes must be continuously maintained. As codes on either side of the interface change, the mappings must be updated to reflect these changes. This much effort must be expended to allow existing applications to be used as organizations begin to integrate their information systems.

In conclusion, what was gained from this experience is a better understanding of the types of tasks and level of effort required integrating information systems. This involved not just the building of an interface but also the determination of the coded values to be used and the building and maintaining of the cross-references needed to support the interface. These tasks will have to be accounted for when planning all future integration efforts.

REFERENCES