An Institution-Based Process to Ensure Clinical Software Quality

Susan A. Abookire, MD, Jonathan M. Teich, MD, PhD, David W. Bates, MD, MSc
Brigham and Women's Hospital, Boston, MA.

Clinical software can have a major impact on the delivery of care. It is imperative that clinical software undergo regular quality review, to evaluate the clinical correctness of the specification, the technical correctness of the software, problems that have arisen, and maintenance of the software as conditions change. We have developed a process using existing hospital review groups to perform clinical review, and using a project specification form and analysis of likely problem areas to effect technical review.

INTRODUCTION

In the past several decades, there has been a rapid expansion of clinical software systems within institutions that deliver medical care. An increasing amount of software is used to assist physicians in making diagnostic and therapeutic decisions. As healthcare becomes more complex, software has a growing influence on diagnosis, treatment, and patient outcomes. The spectrum of impact ranges broadly from look-up tables of drug dosage information, to software that directly monitors patient variables and adjusts therapeutic interventions accordingly.

The Food and Drug Administration (FDA) has regulatory responsibility over medical devices that serve patients. In the 1980's, the FDA began to evaluate the regulation of information systems used in medical care. The FDA requested physician and other expert input to help define its role in this complex arena[1]. In July 1996, a consortium of organizations dedicated to improving health care through information technology met, at the invitation of the FDA, to discuss medical software regulation. In its published findings, this consortium stated a key recommendation that responsibility of assuring quality and monitoring software systems should be placed at the local, institutional level[2].

A two-year project, funded by the National Library of Medicine, was launched to test the feasibility of locally developed, institutionally based software oversight processes at four different institutions including our own. Each institution is charged with the mission of developing a software oversight process, and establishing standards to ensure the safety, efficacy, reliability, and clinical accuracy of software that affects patients. Since there are no existing national standards for clinical information systems, this project has the potential to establish a foundation for an oversight process, which may be shared nationally. This process is also intended to bring to light common themes that can potentially be codified into a checklist of criteria.

In this paper, we describe past work and current developments in the clinical software quality review (CSQR) process at Brigham and Women's Hospital. We begin with a summary of software quality principles and past efforts in CSQR. The next section presents the details of our new CSQR process, along with areas for further development.

ASPECTS OF QUALITY

Brigham and Women's Hospital (BWH) is a 720-bed tertiary care teaching hospital affiliated with Harvard Medical School. The Brigham Integrated Computing System (BICS) supports clinical, administrative, and financial information needs of nearly all hospital departments. BICS runs on a client-server network of over 7000 workstations located throughout the hospital and in separate ambulatory care facilities.

BICS offers a very wide spectrum of clinical information, and includes a large variety of alerts, reminders, and other clinical decision-support processes that can influence the process of care at BWH. All inpatient orders are entered into BICS directly by clinicians. Each day, approximately 400 of the 14,000 orders are changed as a result of active suggestions by the computer[3]. Studies have shown that BICS decision support functions have a significant effect on reducing serious medication errors[4]. Because of the potent impact of BICS on the process of care at BWH, we are naturally concerned with the possibility that errant or clinically inappropriate software could affect care for the worse.

The concept of *clinical software quality* is multi-faceted. It does not relate simply to the accuracy and robustness of the computer program in executing a specified function (technical correctness), but also to the clinical correctness and appropriateness of the specification itself (clinical correctness). This is especially true for clinical decision support interventions that recommend alternative therapies or diagnostic methods. It is possible for a clinical group to suggest a computer intervention without being aware that it could actually cause adverse effects in other situations. For example, a group may specify that a computer recommendation can not be overridden in any circumstance, or may specify that an intervention apply to an exceptionally large percentage of patients.

1091—8280/99/$5.00

Such specifications may be valid in some circumstances, and the exact appropriateness of the circumstances is a matter to be decided by clinical leaders, not informaticians. However, it suggests that most if not all such specifications should receive organized clinical review. Informaticians who have experience in advanced clinical systems are often able to recognize when a specification may possess hazardous features, and to identify the staff and patients most likely to be affected by such a computer process. As well as technical and clinical correctness, clinical software quality also includes monitoring and maintainability, so that the software continues to work correctly, even when clinical knowledge changes or when other parts of the system change. In one documented instance, a surveillance program designed to detect potential cases of tuberculosis generated too many alerts because the laboratory adopted a new test method with a different test code[5]. Trend monitoring enabled the developers to discover the problem quickly. A clinical knowledge base must be maintained, and kept current with clinical updates occurring as frequently as experts deem appropriate. We recognize the need to ensure that initial project design include designation and specification of a software maintenance plan for the project after its development.

A clear-cut mechanism for assembling and incorporating user feedback is another cornerstone to developing well-received, usable software. The conduit for sending feedback ought to be readily apparent to the user, and such feedback should lead to rapid response and follow-up.

Correctness	Extent to which a program satisfies its specifications and fulfills the user's mission objectives
Reliability	Extent to which a program can be expected to perform its intended function with required precision
Efficiency	Amount of computing resources and code required by a program to perform a function
Integrity	Extent to which access to software or data by unauthorized persons can be controlled
Usability	Effort required to learn, operate, prepare input, and interpret output of a program
Maintainability	Effort required to locate and fix an error in an operational program
Testability	Effort required to test a program to ensure that it performs its intended function
Flexibility	Effort required to modify an operational program
Portability	Effort required to transfer a program from one hardware configuration and/or software system environment to another
Reusability	Extent to which a program can be used in other applications; related to the packing and scope of the functions that programs perform
Interoperability	Effort required to couple one system with another

Table 1. Definition of software quality factors.

Software based on a clinical specification is in turn held to a standard of technical quality, which encompasses fundamental factors common to all projects[6,7,8,9]. Basic factors relating to all software have been catalogued previously and are shown in Table 1[10].

APPROACH

A variety of procedures have been successfully applied to past projects on BICS, both prospectively and retrospectively, to detect and prevent technical or clinical quality failures. However, these procedures have not been universally applied, and there has not been a mechanism for effectively transferring procedures applied in one project to another project. The BWH approach to CSQR revolves around methods of guaranteeing appropriate clinical and technical review, and development of an effective method for transferring CSQR knowledge to new projects.

New project life-cycle

In the BWH approach to CSQR, we have adapted and codified existing successful procedures. CSQR is introduced in specific procedures at a number of steps in a project's life cycle:

1) The project initiator or champion develops a draft specification using a structured form that we have developed.

2) The project is reviewed by information systems for general feasibility and likely trouble spots.

3) A clinical *knowledge domain committee* reviews the proposal for clinical correctness and side effects, and documents due diligence for testing, monitoring, and maintenance.

4) The project design team and an IS review group familiar with CSQR perform *hotspot analysis* to determine the likely areas of software failure and the appropriate remedies that must go into the design.

5) An independent body, the software oversight committee (SOC), reviews and monitors the overall CSQR process.

Progress of the CSQR process is tracked by a project document, which serves to convey and categorize the ongoing project specification, and to verify that all of the necessary steps have been performed.

Project initiation and specification

A crucial first step is a thorough project specification. The project initiator (usually a clinician or hospital committee) will identify not only the software objective, but also the underlying clinical goal. By clearly defining this, the optimum intervention and delivery approach to accomplish this goal can be determined. For example, a clinical goal may be to reduce the number of plain abdominal films ordered in the diagnosis of an acute abdomen, based on clinical evidence that these radiographic tests are not useful in this context. The initiator may request a simple alert to physicians who have ordered such a study. However, accumulated knowledge about how computer interventions modify behavior might suggest that a different approach is more effective, such as providing a pull-down menu option of suggested procedures for this situation. By specifying the underlying clinical goal, others in the project chain can apply this knowledge as appropriate. Also, measures by which to monitor the effectiveness of the software tool can be specified and implemented from the start.

We have developed a series of questions to be answered by the individual(s) initiating a new project, resulting in a draft project specification that encompasses the project objective, suggested approach, clinical background, target population, suggested monitoring measures, and parameters by which to evaluate the project effectiveness. A list of these questions, and a sample of possible answers is provided in Figure 1.

Project review

The draft proposal is then delivered to an analyst of manager in our information systems department for a screening review. At this stage, key tools needed to implement the proposed project are identified. A determination is made as to whether existing software may serve as a template for proposed software, or whether new techniques and paradigms are called for. Feasibility, scope and scale are assessed.

TITLE: Flu shot reminder system

1. **OBJECTIVE**: increase the number of eligible patients who get flu shots in our outpatient practices

2. **APPROACH**: determine eligible patients who have visits today; put reminders to give flu shots on the standard printed schedule sheet

3. **CLINICAL BACKGROUND**: flu shots should be offered to eligible patients. In the past, mailings to patients have not been effective at increasing flu shot use. Many patients have not received flu shots even though they had a regular visit with their provider.

4. **SELECTION CRITERIA**: use CDC guidelines to determine what patients are eligible for flu shots, and duration of season in which to offer them.

5. **EXCLUSION CRITERIA**: patients whose health grid shows a flu shot for the current flu season, or who have an egg allergy

6.**TARGET POPULATION FOR INTERVENTION**: primary care physicians and nurses in outpatient practices.

7. **USER INTERFACE**: printed suggestion to offer flu shot, if appropriate, at bottom of schedule sheet.

8. **MONITORING**: assess if patients who meet eligibility criteria do get printed reminders; monitor proportion of patients getting flu shots.

9. **EVALUATION**: analysis of proportion of eligible patients in practice who receive flu shots.

10. **PRIMARY STAKEHOLDERS**: directors of ambulatory practices.

11. **CLINICAL CHAMPION FOR THIS PROJECT**: Dr. Jennifer Smith.

12. **URGENCY / REQUIRED DELIVERY TIME:** Before September 1.

13. **WHOSE JOBS DO YOU EXPECT TO BE AFFECTED BY THIS PROJECT?** Practice managers or secretaries who print up schedules; providers; nurses or assistants who administer flu shots.

14. **WHAT ARE POSSIBLE ADVERSE CONSEQUENCES OF IMPLENTING THIS PROJECT**? What if the reminder is given on a patient who had a flu shot already (elsewhere, or here but after reminder was queued or printed)—would patients receive extra flu shot?

Figure 1. Project Specification Sample.

Clinical Review

We have identified seven knowledge domain committees (KDCs), each with an area of expertise (listed in Table 2). Most of these are based on pre-existing hospital committees, and provide an array of expert domains. We have tested the completeness of our KDC list by allocating, in theory, all existing software projects to one of them and have found them reasonably complete. In the future, this list may need expansion or further cross-referencing of domains.

A representative of the project initiation group presents the draft specification to the appropriate KDC for a full vetting of the project specification. Key stakeholders, previously identified, as well as affected employees from other areas are represented. A member of the proposed target population is invited. The project is presented and discussed. Either a consensus on the final specification is achieved, or the initiation team must iterate the proposal based on feedback from the KDC.

Pharmacy & therapeutics
Laboratory department
Radiology department
Ambulatory care improvement group
Health promotion and education committee
Clinical communication committee
In-patient advisory group

Table 2. List of Knowledge Domain Committees.

Development phase

A project development team then receives the consensus specification. The development team performs hotspot analysis, and prepares plans for testing, training, and monitoring, based on the specification.

Hotspot analysis (Figure 2) is based on a categorized collection of past problems and solutions in clinical software at BWH. The initial set of hotspots was derived from feedback logs, problem logs, and interviews with analysts and developers. We catalogued this set into a database, sorted by program feature. By doing this, designers of any other software project that uses a similar feature will be attuned to specific problems associated with the feature. For example, any program that sends critical alerts by automatically paging a physician must know what to do if the page system is down, or if the physician's pager is off. A recommended remedy is to require a response from the physician, and to escalate the alert to a different person or a different technology if no response is received after a certain length of time.

To optimize our accumulated knowledge of hot spots and key problem areas, we plan to continue to capture all reported problems into the hotspot database. An analyst will catalogue these into a growing guide of error types, software types, impacts, and solutions. While this may be labor-intensive at first, we believe that pooling and referencing this knowledge is a first step toward eliminating error repetition caused by a lack of knowledge sharing

A systematic test plan must be developed, approved, and implemented. Goals of the test plan must include 1) that the specifications are met (software correctness), 2) that the system is stressed (problem discovery is undertaken), 3) usability is evaluated, and 4) failsafe methods are implemented and exercised during testing. Individual modules of the software are separately tested

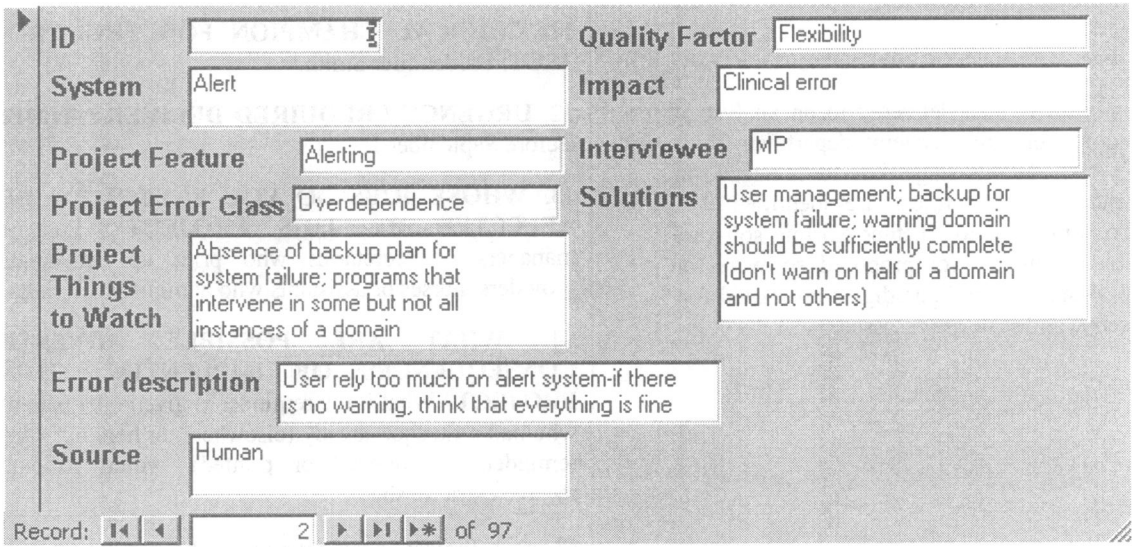

Figure 2. One record in the hotspot database.

for problem isolation, and overall function from input to output is also systematically tested to allow for complex exercising of the software.

The information systems review group is a small group of persons who participate in the review process on all projects. At this review, these persons can lend their CSQR experience to the benefit of each project. In addition, an evaluation of the quality proposal and test plan is done.

Software Oversight Committee

Our software oversight committee, an independent group which functions asynchronously with the rest of the project flow, performs ongoing monitoring and evaluation of our process. This committee, drawn from the medical staff, nursing, administration, information systems, and quality assurance, is charged with the mission of independently monitoring our process and determining its effectiveness. Our committee meets every other month. This group has a mission to evaluate the CSQR process as a quality improvement effort. As such, it does not monitor each project in detail, but ensures that due diligence is applied to each and the process is upheld. Lessons learned from KDC meetings and information systems review are shared in this group as a means of improving the process and fostering communication between groups. Further, our SOC has the mission to provide an independent perspective, ensuring that our clinical information systems continue to meet organizational goals.

DISCUSSION

At this stage in our project development, we have formed or empowered the above committees, developed the hotspot database, and are beginning to run specific projects through the process. Evaluation will include determining how projects are affected and how much work is added, or saved later because of reduced problems down the line. Other measures will include the number of problems caught in the process, and the number not discovered until implementation takes place. The overall costs and benefits of this process must be assessed. An ultimate issue is whether such internal review is sufficient or whether outside regulation would be useful in addition. While we believe the former, this remains an important policy issue.

As this multi-site project continues, we will continue to share our evolving understanding and lessons among the sites. Because of the diversity of sites, we are inherently testing the feasibility of having local, unique institutions develop their own CSQR process. In addition, as we learn from each other's successes and failures, we hope to gain a common knowledge which will apply to various types of institutions, with an eye toward establishing a guideline of fundamentals and tools to help institutions develop their own clinical software quality assurance process.

We gratefully acknowledge the support of the National Library of Medicine, grant #1 R01 LM06591-1.

REFERENCES

[1] Young, Frank E. Position Paper: Validation of Medical Software: Present Policy of the Food and Drug Administration. Ann Intern Med 1987; 106(4):628-629.

[2] Miller RA, Gardner RM. Summary recommendations for responsible monitoring and regulation of clinical software systems. Ann Intern Med 1997; 127(9):842-5.

[3] Teich JM, Glaser JP, Beckley RF, Aranow M, Bates DW, Kuperman GJ, Ward MA, Spurr CD. Toward cost-effective, quality care: The Brigham Integrated Computing System. In: Steen (ed.), Proc. 2nd Nicholas E. Davies CPR Recognition Symposium. Chicago: Computer-Based Patient Record Institute, 1996; 3-34.

[4] Bates DW, Leape LL, Cullen DJ, Laird N, Petersen LA, Teich JM, Burdick E, Hickey M, Kleefield S, Shea B, VanderVliet M, Seger DL. Effect of computerized physician order entry and a team intervention on prevention of serious medication errors, JAMA 1998; 280(15):1311-6.

[5] Hripcsak G, Knirsch CA, Jain NL, Pablos-Mendez A. Automated tuberculosis detection. J Am Med Inform Assoc 1997; 4(5):376-81.

[6] Murfitt RR. United States Government regulation of medical device software: a review. J of Med Eng & Tech;14(3):111-113.

[7] Cosgriff, P S. Quality Assurance of medical software. J of Med Eng & Tech; 18(1):1-10.

[8] Sailors RM, East TD, Wallace CJ, Carlson DA, Franklin MA, Heermann LK, Kinder AT, Bradshaw RL, Randolph AG, Morris AH. Testing and validation of computerized decision support systems. J Am Med Inform Assoc 1996; 3(5, suppl.):234-238.

[9] Schulz EB, Barrett JW, Price C. Read code quality assurance: from simple syntax to semantic stability. JAMIA 1998;5(4):337-346.

[10] McCall JA, Richards PK, Walters GF. Factors in software quality. Rome Air Development Center, RADC-TR-77-369, November 1977.