CHAPTER

Features of Computer-Based Clinical Decision Support

3

Robert A. Greenes

The purpose of this chapter is to dig deeper into the nature of computer-based clinical decision support, in terms of the ways in which it is or potentially could be used, its design, and its interaction with host environments. I believe that understanding these aspects of CDS is important as a foundation for serious efforts to increase its dissemination and adoption. As pointed out in previous chapters, and as will be explored in more detail in Section II, much of the success of CDS has been with one-off implementations that have been difficult to maintain over time even within their own institutions, more problematic when extended for use throughout a health care enterprise, and only rarely replicated elsewhere despite demonstrated effectiveness. If we are to tackle this issue and break down barriers to dissemination and adoption, we need to know what we are working with, which aspects are most troublesome and how they can be improved, and which components or interfaces can be standardized or made easier to deploy. We also need to understand the human factors and process and workflow implications of CDS use, so that we can determine optimal approaches to invocation and user interface design.

An underlying thesis is that CDS has a *conceptual architecture* comprised of a number of design elements or components. Many kinds of CDS are designed without the architecture being explicit, but I will try to demonstrate that all the design elements nonetheless are present, even if implicit. A second thesis is that CDS does not function in isolation, but rather that it *operates* in the context of some sort of *application environment*. Nonetheless, in the discussion to follow, I will consider the chunk of software that provides CDS as a "module" of software that interacts with the application environment, recognizing full well that often CDS implementations are not modular at all, and that the dividing line between what constitutes CDS and what constitutes the application that invokes it is frequently not clearly defined, so that the two cannot be cleanly separated. In any case, we can think of the functions that are performed as being done by either the application environment or by the CDS software, so that responsibilities for these functions can be assigned to one or the other.

These two conceptual idealizations – 1) an underlying component architecture for CDS, and 2) modularity of CDS, in that its tasks and responsibilities can be separated from those of the application environment with which it interacts – are

Clinical Decision Support.

helpful to better understand how to design CDS in a portable, reusable, maintainable fashion. I believe that, in adopting this approach, it will also be possible to design more robust versions of existing CDSs that have the ability to interoperate in other platforms, be adapted to differing application settings, be maintained and updated more readily, and thus be more widely disseminated and used.

So this chapter is about design principles. But as you will quickly conclude, this topic is a work in progress. There are many complexities, nuances, and unresolved questions yet to be answered by researchers and developers in this field. My goal is to call attention to these design principles and challenges as a framework for further work, in the belief that this will help to accelerate our progress in dissemination and adoption.

3.1 CDS and the human

A central characteristic of CDS is that it is intended to interact with and give advice to a human being. Our focus has been primarily on clinical decision support to health care providers, generally physicians and nurses, but also sometimes pharmacists, technologists, and other personnel. Many of the same principles apply to patient-centered decision support, with its added complexities of health care literacy, language, and mental models. Sometime CDS will involve processes for shared decision making between a health care provider and a patient, family member, or other caregiver. We discuss patient-centered decision support in Chapter 27.

The ability to provide advice to a physician has in many ways been a *disruptive innovation*, to borrow a phrase from the business world (Christensen and Raynor, 2003), in terms of traditional perceptions by physicians of their roles and responsibilities, and the practices and relationships that derive from those perceptions. Over the years I have collected cartoons clipped out of magazines and journals, portraying the use of computers in health care. Almost all of these relate to some sort of role of computers or information technology in making decisions. This focus no doubt exemplifies the way most people first think about computers if asked to consider their potential role in health care.

In one cartoon, a patient is consulting a computer and the computer is advising, "Take two interferon tablets and call me in the morning." In another, several surgeons around an operating table are discussing an operation they are about to perform. One is saying to the other, "Since it's been reported that 24% of surgery is unnecessary, let's only do 76% of the procedure." In yet another, a patient is entering a question into a computer and the response is "Not tonight – I have a headache." Another cartoon with an operating room venue shows a surgeon with two hearts, one in each hand, looking across the table at his colleague and saying, "Okay, the old one is in my left hand and the donor's is in my right, correct?" A final cartoon shows a doctor examining a patient. The patient has an arrow in his back, unseen by the doctor. The doctor says, "I'm pretty sure it's psychosomatic, but let's run some tests to be sure."

The humor in these cartoons has to do with the exaggeration of two opposing views of the relationship between the computer and the human, whereas a suitable relationship likely lies somewhere between these extremes.

3.1.1 Computer as omniscient sage

On the one hand, an extreme view is portrayed in which the computer's expertise is taken for granted, and its pronouncements are *accepted quite literally*; the computer is essentially *running the show*. The popular image of computers in medicine is that they are devices that are capable of storing vast amounts of information, performing lightning-fast computations, and making accurate decisions. The cartoons are funny because they start from that assumption and carry it to an absurd extreme. The computer may be seen as patronizing and even arrogant.

3.1.2 Computer as out-of-touch meddler

The opposite extreme is the traditional view of resistance by physicians to computers and to automated guidelines as being too simplistic and representing *"cookbook medicine*" – with the typical warning that computers are insensitive and incapable of recognizing the nuances of patient care, the role of physician judgment, and the prerogative of the physician as primary decision maker.

3.1.3 A more symbiotic view

In actuality, the use of computers in health care has taken a rather conservative, circumspect, and circuitous route to participating in clinical decision making. That route is clearly between the two extremes of the human ceding control to the computer and the human not being willing to use the computer for decision support at all. Later in this chapter, we discuss a number of dimensions along which the nature of the human-computer interaction in CDS can be considered, including locus of control and degree of assertiveness. The extent to which the interaction is skewed in one direction or another along either of these or other dimensions will depend on the application and purpose, but it will rarely be entirely in one direction on all dimensions. As will be exemplified in a number of case studies and historical reviews in Section II, when CDS has been successfully deployed, the computers usually have been used primarily in an advisory or educational role, in providing input to the practitioner or patient, who ultimately is responsible for making all decisions. This is why, when we discuss CDS in this book, it will largely be with that perspective – in other words, the emphasis is on decision support, *not* on decision making.

I mentioned earlier that CDS is a sort of disruptive technology. One manifestation of this is that there has been a change in patient attitudes toward their doctors, both in recognizing the limitations of knowledge and judgment of physicians as a group, and in an increased tendency to question a physician's decisions and desire

or demand to participate in the decision-making process. Further, whereas in the past, physicians were reluctant to consult an information source such as a textbook or a computer in front of a patient for fear that it be regarded by the patient as a sign of indecisiveness or lack of knowledge, there is growing evidence that the ability to look up the latest information is regarded by both patients and doctors as necessary and desirable (Ogden et al., 2002; Weaver, 2003).

Only in limited circumstances in clinical medicine might one consider using the computer directly in a closed-loop fashion to collect data, analyze the data, make decisions, and take actions without human intervention. Probably the most notable exception is the implanted electrical cardiac pacemaker, which has algorithms for determining when to stimulate the heart automatically in response to heart rate or rhythm abnormalities. Patient ventilators can do some automatic manipulations, such as use of feedback control to adjust cycling thresholds to maintain a desired pressure level, adjustments to keep PEEP/CPAP pressure at specified levels to compensate for gas leaks, or modification of ventilator flow delivery rate to adapt to changes in patient inspiratory effort. However, any such use requires considerable caution and documentation of efficacy, and is done only after an arduous process resulting in regulatory approval from the Center for Devices and Radiological Health of the Food and Drug Administration (FDA) in the United States (US) (Hackett and Gutman, 2005), or from comparable agencies in other nations (Altenstetter, 2003).

In most situations, the human remains in the middle of the decision-making loop. The guiding settings on a ventilator are still determined by a human, after viewing data obtained from the device and other information such as blood gas test results. Insulin infusion pump settings are still adjusted by humans, after reviewing laboratory and clinical data, even though the device could presumably have an algorithm for automatically responding to the most recent blood glucose laboratory result in the EHR. For clinical decisions that are not integrated with embedded or connected devices, recommendations are not implemented without the express approval or action by the human.

3.1.4 Limitations of the technology

Given our present state of the art, computers can usually be expected to provide only relatively unsophisticated decision support, and computer models of human decision making remain limited. This relates primarily to two factors. First, many kinds of data and nuances regarding patient findings are either not captured by a computer or are encoded in a form that the computer can interpret, but which an experienced clinician not only has access to but can use more effectively. Second, some of the nuances of the decision process that could potentially be captured may not even have been encoded in the model used by the computer, but which a physician routinely considers. Because of the limitations of both the data available to the computer and the model used, it is thus important for the physician to check the reasonableness and appropriateness of a CDS recommendation before acting on it.

A further difficulty is that some of the applications of CDS that have been pursued are quite complex, for example, determining a differential diagnosis, deciding on an optimal work up strategy, and doing treatment planning. Thus it is not surprising that most of the success to date has been in the form of simpler kinds of CDS, where the modeling of the decision problem and capture of the nuances of data are much less challenging. Examples of such kinds of CDS include the use of single-decision rules in targeted settings, such as in CPOE for providing checks of medication doses against recommended ranges, or verifying the absence of allergies or recognized drug interactions; or in checks of results of newly arrived laboratory results against normal ranges in order to alert physicians of abnormalities. Though these are considerably easier to implement, even they have subtleties and nuances that must be considered for successful implementation. For one thing, the simpler the rules, the less they take into account a variety of mitigating factors that affect the clinical significance of the potential recommendation. But increasing the patient-specificity and sophistication of the rules requires more data, and of course access to the EHR. Also, to avoid redundant alerts and the well-known problem of "alert fatigue" (see Chapter 1), the logic should take into account past history of the condition and also check whether a similar alert has been generated within a specified "alert-fatigue avoidance" time window. Accounting for such factors makes the rules and the maintenance of them much more complicated, as a result of which the rules are no longer simple (Lee et al., 2010; Embi and Leonard, 2012; Kawazoe et al., 2013).

Last, replication of some of the approaches shown to be successful in early-adopter settings often has been problematic for a number of other reasons beyond the decision model and the availability of the data. Factors have been both technical, cultural, and organizational in nature, which we will discuss in greater detail next, and in subsequent chapters, particularly in Sections IV and V of this book. For example, as discussed briefly in Chapter 1 and at the beginning of this chapter, when we consider the use of computers in interaction with human beings, an important issue that needs to be carefully addressed is how that interaction is regarded by the human user in terms of decision-making control, responsibility, and judgmental prerogatives. Other factors include the manner of interaction of the program with users in terms of impact on their ease of performance of operational tasks, time required, and effect on workflow procedures and processes, particularly as they relate to clinical IT services.

As we pointed out in Chapter 1, another reason for the gap in adoption has to do with underestimation of the complexity of the tasks involved in replicating an innovation such as CDS on a widespread basis, given the need for it to be well integrated with clinical information systems, actual workflow, and business and health care practice patterns in each site; and given the need for it to be readily updated and adapted to changing requirements. Challenges in deploying CDS at each site are introducing it in a way that is acceptable to the individuals who will be required to use it, being sensitive to the culture, work style, time constraints, self-image, and other cultural and social factors of these individuals and the organizations in which they participate.

3.1.5 Considerations regarding human-computer interaction

Computers have long been regarded ambivalently as both a boon and a possible threat with respect to their interaction with humans, particularly for decision making. The field of *artificial intelligence* was specifically created more than 50 years ago with the aim of exploring the nature of intelligence, including the processes of acquisition and representation of knowledge and the ability to carry out reasoning and problem solving (Feigenbaum and Feldman, 1963). This has involved both research studies and demonstrations centered on how to make intelligent computers as autonomous entities. We see a number of applications of this kind of pursuit in the form of advances in robotics, chess playing and other strategic game-playing programs, speech recognition, and automatic language translation. A notable recent example was the challenge by Watson, an IBM software system, against three champion players on the popular television show *Jeopardy!,* in which the task is to state a correct question for which a trivia fact is provided as an answer. Needless to say, Watson defeated the humans, relying on vast storehouses of facts and relations, accumulated from text processing and clever programming (IBM Research, 2013).

The problem of trying to build intelligent autonomous computers is a fascinating one, but this has less direct applicability in health care than the use of computers in partnership with humans. Many of the same methodologies are used as in the pursuit of autonomous intelligent computers, but an additional focus is on the nature of the interaction between the computer and the human (Johnson, 1994). Our concern here, as we have noted, is the role of the computer as a decision-*support* tool rather than as a decision-*making* entity.

What is the best way in which an "intelligent" computer should interact with a human to provide CDS? There are several possible modes of interaction: First, a computer can be *in charge* of the interaction, delivering recommendations or decisions that are expected to be carried out, and at the right point of time. This mode could be used, for example, in CPOE, when an attempt is made to order a medication with a dosage that is outside of therapeutic limits for safety. The computer should be able to either actively stop such orders from being processed or passively avoid them by not providing the means to enter (or select) such doses in the first place. In another sense, however, the choice of ordering the medication is still made by the physician, and it is typically only when the entered or attempted order needs to be overridden by the computer, because the dose is outside of therapeutic range, that the computer exerts control of the interaction.

A slightly less assertive version of computer control is a mode in which a human decision maker can override the computer by providing a justification for an action to which the computer has raised a warning. An example of this mode of use in order entry systems is when a procedure is ordered by the user for an indication that is not recognized by the computer as being among those generally accepted, but which is nonetheless permissible (Harpole et al., 1997).

Relaxing the constraints still further, consider the mode in which a computer presents a data entry form or dialog box to be completed by the user. The entries

in the form are checked for validity, e.g. to ensure that they are within range of values expected for the requested items or that they match a controlled list of possible entries; or the computer may require that entries be chosen from a drop-down list. Thus the kinds of allowed input are controlled by the computer, but within that scope, the user may make any desired choices. Following entry of data, subsequent displays or forms can be made available to the user, and the sequence and nature of the interaction guided by the computer in response to user entries. An example is a predefined order set for medications and procedures, in which a physician might be interacting with the computer to order most of the procedures but also to customize some of the options. Standard dosage forms of medications might be offered, both to encourage those choices and to make it faster for the user, if he or she chooses them, but still allow entry of alternative dosing regimens.

If we consider shifting the focus of decision making further in the direction of user control, several modes of interaction are possible. In one mode, the user performs various actions, and the computer analyzes them in the background, displaying a warning when the action is considered to be dangerous or inappropriate. Among the earliest experiments aimed at refining this approach was a series of studies carried out by Miller and colleagues (Miller, 1983; Miller and Black, 1984) on "critiquing systems." As discussed in Chapter 2, the primary applications investigated were in management of hypertension and anesthesia. In the critiquing mode, the computer made comments and recommendations for modifying notes and orders already created by the physician, which the physician could accept or reject. A more contemporary application is the typical circumstance in CPOE in which a physician is able to select choices for medications and doses directly, but the computer identifies some of the chosen orders or doses as contraindicated in this patient because of interactions or allergies or inappropriate dose (Kuperman et al., 2001), and advises the user of alternative actions that may be more appropriate.

A somewhat more passive mode of interaction is one in which the computer gathers statistics about the performance of users over some period of time and provides feedback to the users about how they compare with their peers. This can be regarded as more an educational type of intervention than as CDS unless it is provided in a highly patient-specific context. It has been shown to have mixed success, and appears to be most effective when coupled with a concerted educational initiative and buy-in by the physicians of this kind of decision support (Mugford et al., 1991, Bindels et al., 2003; Bodenheimer, 2003; Greenhalgh et al., 2005).

One of the primary targets for concerns about CDS invading a physician's autonomy is the use of clinical practice guidelines, especially those that are computer-based. Although clinical practice guidelines have abounded in magazines and journals, been distributed on CD-ROMs, and are maintained on Web sites, they are rarely used in clinical practice by physicians, except as education or reference resources. In the care of a specific patient, the experience is often that the guideline does not capture the nuances of the patient, and/or does not embody what the physician believes to be best practices in his or her institution or in the current setting or to correspond with his or her own experience. There is some justification in these

complaints. Clinical practice guidelines, however comprehensive they may be, usually cannot specify the details of every possible combination of circumstances that might arise in practice. If they could do so, nonetheless their rendering in a print or display medium for easy comprehension and use would be a significant challenge. Even for standardized guidelines such as those for hypertension management (NHLBI, 2004), a flow chart rendering of the various alternative pathways for management would take up many pages.

Another problem with clinical practice guidelines is that many of the characteristics of patients may be outside the scope of the guideline, for example, other concurrent diseases or medications that can alter the nature of the current condition; or the presence of findings that are not available to the computer-based guideline, for example, those that relate to nonverbal subjective assessments that can best be made face-to-face and are difficult to articulate in words. Finally, even the most well-researched, evidence-based guideline may not have achieved 100 percent consensus among experts, and alternative modes of care may exist that would be equally appropriate. Thus, guidelines can be best used in a mode in which they provide suggestions or advice when requested, but do not force compliance.

Due to limitations such as those just cited, the idea of clinical guidelines tends to conjure up some of the worst associations with the term "cookbook medicine" (Liang, 1992; Harding, 1994; Costantini et al., 1999) that we mentioned at the outset of this chapter. At their base, the objections relate to the view that medicine is too complex to be reduced to a set of algorithms or rules, and that it could never be codified to an extent similar, for example, to that which enables an autopilot to fly an airplane. But, in fact, no one is advocating the autopilot as a model for computer-based decision support in health care. Autopilot operation is successful in airplanes largely because the procedures and operations of normal flight are highly predictable, based on data that can be objectively gathered. As a result, rules for decision making can be fully specified and implemented. Autopilot systems also can be made "aware" of settings in which their use is not appropriate (e.g. takeoff and landing) and circumstances in which something happens that is outside of their realm of decision making, such as the occurrence of a combination of parameters for which there is no defined rule. In such situations, either automatically or through pilot initiative, they have a mode in which the pilot can take over control or override their operation. In the autopilot setting, the computer is more in control than the human most of the time, but the initiative can switch to the human. There are situations in health care that can approach this, for example, the aforementioned closed loop systems of implanted cardiac pacemaker devices, or other applications that potentially could become semi-automated, such as intravenous infusion systems for medication administration or patient ventilator management systems for adjusting O_2 levels or cycling thresholds of the ventilator. Also, guidelines and protocols can be used by physician assistants and nurse practitioners to deliver care in routine situations such as management of sore throat or routine prenatal care, where there is also a clear understanding of when to escalate the care to a more experienced provider.

In general, the cookbook response stems from misconceptions regarding the ways in which clinical practice guidelines can be useful in CDS. Computer-based guidelines can be made more patient-specific in several ways. First, they can be freed of the constraints of paper or screen display of static algorithms, by supporting the identification of a variety of possible entry points and the eligibility/applicability criteria for these entry points, and by supporting more flexible means for browsing and navigation of pathways and access to explanatory materials (Abendroth and Greenes, 1989). Second, guidelines can be made arbitrarily more complex and nuanced with respect to patient findings (subject to limitations on available evidence, author expertise, and author fortitude in delineating all of these circumstances), since the need to render the guideline in static form is no longer an issue. Third, the guideline can be decomposed into parts that can be deployed in various application settings, such as those parts best used during CPOE, those that would be most appropriate as alerts or reminders, or those that should be considered during the patient assessment and progress-note-generation process (Essaihi et al., 2003; Wang et al., 2003). Fourth, these parts can be specified precisely so that they can operate on entered or stored EHR data and produce their recommendations automatically (Tu et al., 2004). The issues involved in automating clinical guidelines are discussed further in Chapter 13.

We will return to the general issue of interaction with the user when we discuss the process of integrating CDS into application environments later in this chapter.

3.2 Design and structure of CDS

Many opportunities exist for performing CDS. Two reviews in the early to mid-2000s have developed taxonomies of features (Sim and Berlin, 2003) and modes of use (Kawamoto et al., 2005), respectively, for clinical decision support. The Kawamoto study (Kawamoto et al., 2005) and a follow up to the Sim study (Berlin et al., 2006), in particular, are noteworthy in identifying those forms of CDS that have been evaluated in clinical trials. Such schemes, as they are further refined, can be expected to be helpful in continually evaluating instances of CDS in terms of their focus and the settings and modes of their deployment to determine which are most effective.

In this section we propose our own schema for considering CDS, in terms of (1) its purpose, and (2) the architecture and component design elements required for providing it. Design elements include the decision model, knowledge content, data requirements, result specifications, and application environment factors affecting deployment and use. Since one of our main objectives is to understand what factors have held back widespread adoption of CDS and to identify ways of increasing this adoption, we will analyze aspects of this schema to help us answer these questions. The questions to be addressed encompass the kinds of standards and infrastructure that may be needed, the kinds of business and organizational strategies that can be useful, and the kinds of approaches that can be used to encourage wider adoption.

3.2.1 Purpose

We now turn to a consideration of the many possible goals or purposes for which CDS might be intended (Table 3.1). Purpose is somewhat orthogonal to the classification of methodologies we described in Chapter 2. Thus, as we consider the various purposes for CDS, we identify the key methodologies that have been used for their implementation. Since references to those methodologies are cited in

Table 3.1 Principal purposes for CDS, and the key methodologies used

Purpose	Key methodologies
Answering questions	Direct hyperlinks from context-specific settings, context-specific information retrieval, use of agents and information brokers, infobuttons as instance of the latter, or ultimately, a "personal guidance system"
Making decisions	Gathering data, analyzing the data, and providing recommendations for assessments or actions
• Diagnosis	Bayes' theorem, algorithmic computation, heuristic reasoning, statistical data mining/pattern recognition methods
• Test selection	Decision analysis, logical rules/appropriateness criteria, and logistic models and belief networks for risk prediction (e.g. for screening decisions)
• Choice of treatment	For choosing among alternatives, decision analysis, and logical rules/appropriateness criteria, including increasingly genotype considerations. For dose modifications for age or factors such as renal function, algorithmic computation. For dosimetry or dose distribution, algorithmic computation based on geometric and pharmacokinetic models, with use of heuristics and statistical methods for optimization
• Prognosis	Logistic regression, Markov modeling, survival analysis models, and quality of life assessment scoring methods
Optimizing process flow and workflow	Multistep algorithms, guidelines, and protocols, coordination of participants by workflow modeling, scheduling, and communication methods
Monitoring actions	Use of ECA rules, with background detection of events, in real-time or asynchronously, logical evaluation of conditions, and issuing of messages. Events can be a user activity such as choice selection or data entry, a result arrival, or the passage of time
Focusing attention and enhancing visualization	Organization and presentation of items in data entry, display, or reporting applications. May be done by use of sequences to encourage intended behaviors, by a process flow model such as an underlying guideline, and/or by visual groupings based on shared attributes such as purpose, medical subdomain, or application context. May also include dashboards, trend plots, or other summarization and visualization methods that make it easier to identify key elements needed for decision making

Chapter 2, we won't repeat them here, but we do cite other aspects of methodology or examples of use not fully discussed in that chapter.

3.2.1.1 Answering questions

The simplest goal for CDS is to provide context-specific access to relevant information for a human user at the time of problem solving or decision making. Hyperlinks to specific resources at specific points in the interaction with a clinical IT system provide one such way to do this. An example would be a link to laboratory tests and their normal ranges, or to a list of medications in a hospital's formulary.

More sophisticated approaches involve using intermediate search tools, such as information agents or "bots" to go out to diverse sources and report back (like Web crawlers or spiders), and "information brokers," which can map queries to the formats required as input by external knowledge bases and then map the responses to a form recognized by the requesting source. The goal is to find resources relevant to a particular context, including patient-specific parameters. For example, in a lab test result review context, the display of a lab test result might be accompanied by an infobutton (Cimino et al., 2002) that, when selected, dynamically retrieves available resources about the test, such as normal ranges, textbook materials regarding the use and interpretation of the test, information about the diseases in which it is abnormal, and a list of MEDLINE references on the clinical use of the test. As discussed in Chapter 2, an HL7 standard for an Infobutton Manager (Del Fiol et al., 2012) to be able to invoke external resources by context in a consistent manner has been adopted, encouraging use of this approach. One can imagine increasingly sophisticated question answering systems (Yu and Cao, 2008; Del Fiol et al., 2013; IBM Research, 2013; Jonnalagadda et al., 2013) being automatically invoked in these contexts to provide highly specific information for decision making.

3.2.1.2 Making decisions – about diagnosis, test selection, choice of treatment, and prognosis

This purpose, in contrast to the more generic task of finding information just discussed, is specifically for help in analyzing information needed for a decision. This can be for a variety of types of decisions, including making diagnoses, selecting tests, planning therapy, and estimating prognosis.

We have noted that differential diagnosis has been among the uses of CDS that have most captivated interest from the earliest days of computer use in health care. An excellent book edited by Berner (Berner, 2010) focuses largely on diagnostic decision making, and reviews the many approaches that have been pursued. The basic goal of differential diagnosis is to deduce, from a set of findings, the diagnosis that best explains them. This is clearly an important task, not only in order to be able to select proper treatment but also to be able to estimate prognosis and to give advice to the patient.

It should be recognized, however, that diagnosis is not usually a single event, but rather a *process* of continually refining knowledge about the patient by

gathering data, performing tests, and re-evaluating data, until sufficient confirmation is reached in order to take therapeutic action. Some approaches, such as decision analysis, appropriateness criteria, and clinical guidelines, have focused on structuring this process, rather than on the endpoint of diagnosis. Indeed, the decision table approach for representing a guideline (Shiffman and Greenes, 1991), as illustrated in Table 2.3 in Chapter 2, doesn't even choose a diagnosis, but determines next actions based solely on combinations of findings. Issues that must be considered in this view of diagnosis as a process relate to the selection of appropriate tests based on cost, risk, inconvenience, and other factors versus the potential for information gain from the tests. When one considers the fact that the institution of treatment is also a diagnostic test, in terms of providing information about how the patient responds to it, it can be seen that the whole patient care process continually involves diagnosis in the form of ongoing reassessment. Prognosis estimation can also be regarded as a type of diagnostic assessment, in that it characterizes the patient's current state of health as one with a particular expected survival rate and quality of life.

For the purposes of exposition, we can divide the topic of making decisions into methods for hypothesis formation or refinement, both for diagnosis and prognosis estimation, and those aimed at performing an action (e.g. test selection, or choosing or detailing a treatment regimen), as depicted in Figure 3.1.

Diagnosis. The process of diagnosis can be subdivided further into detection and classification. Although some screening recommendations remain controversial, the number of tests in use for screening purposes is sure to increase as a result

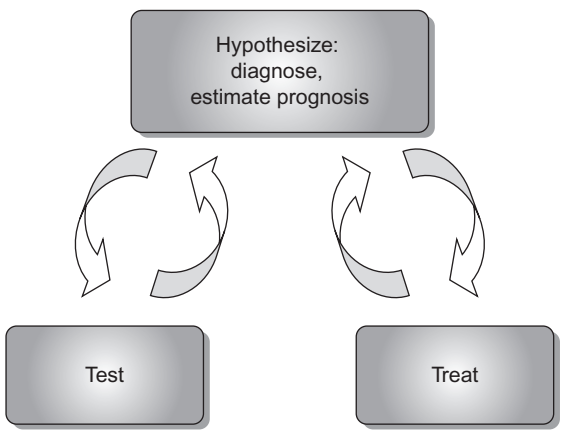

FIGURE 3.1

Medical diagnosis as an iterative process of forming hypotheses and gathering data to confirm or refute the hypotheses. Prognosis estimation is a form of hypothesizing. Treatment is a form of test, in that it also provides data that can help to confirm or refute hypotheses.

of progress in understanding of the genetic basis of disease and development of biomarkers. Screening tests in common use include, among many others, testing for phenylketonuria (PKU) in newborns, mammography in older women or those with certain risk factors, colonoscopy in average-risk patients over age 50, and prostate-specific antigen (PSA) testing in older males. Generally, screening tests have been applied primarily to alert the user to the detection of the possible presence of disease, rather than to make detailed specific diagnoses. The typical approach in screening is to set a liberal operating point (decision threshold) on the ROC curve for considering the test to be positive, on the basis of the view that it is preferable to err in the direction of more false positives than to fail to detect cases of disease (false negatives). Although some assessment tools for supporting CDS in this realm (such as computer-aided detection and diagnosis (CAD) image processing methods in digital mammography (Jiang et al., 1999; Giger, 2004)) try to classify the findings in terms of specific diagnosis, the ability in screening tests to make diagnoses is usually quite limited, and further testing is typically required. A particular caution is raised by Kohane et al. (2006), with respect to whole-person genome mapping, because of the huge potential for false positives and induced tests to further evaluate them, which the authors refer to as the "incidentalome".

Methods used for both detection and classification (diagnosis) include Bayesian probability revision, algorithmic approaches (e.g. for electrolyte imbalance), heuristic reasoning and weighting of findings (e.g. DXplain (Barnett et al., 1987)), and statistical data mining/pattern recognition methods such as logistic regression, classification and regression trees, and artificial neural networks.

Test selection. This set of clinical decision-making problems relates to whether and when to do screening, what test to use for screening or for diagnosis, and determination of need for and selection of follow-up testing, including referral to consultants/specialists. Test selection decision support relies on prior work by researchers in technology assessment including ROC analysis, and comparative effectiveness research (CER) (Sullivan et al., 2013) studies to determine relative performance characteristics of competing alternatives, and work by health policy analysts and payers using clinical decision analysis and benefit-cost analysis to establish the optimal pathways that sequence the use of these in terms of maximizing expected utility, and formalizing these into a set of logic rules or appropriateness criteria.

The decision to obtain screening tests or follow-up tests may be modified on the basis of CDS tools, such as those for estimating a patient's risk of breast cancer based on risk prediction models (Rockhill et al., 2001; Freedman et al., 2005), and policy recommendations/guidelines such as those of the American Cancer Society for early detection of cancer (Smith et al., 2004). A major resource for such recommendations is the US Preventive Services Task Force, which weighs evidence and publishes periodic recommendations on prevention and screening testing (USPSTF, 2013).

Treatment decisions. Needs for CDS include picking the most appropriate therapy, and determining dose or dosage administration regimen. Picking therapy

involves trade-offs of cost, risk, benefits, and patient preferences. Thus, as in the case of test selection, treatment recommendations can be developed by researchers, including economists and policy analysts who rely on comparative effectiveness research, decision analysis, survival analysis, and benefit-cost analysis. Optimal strategies may be issued by professional societies and payers, and codified as logic rules and appropriateness criteria, or in the form of clinical practice guidelines.

For dose determination, CDS can be used to constrain choices, make it easier to select preferred choices through displayed groupings such as order sets, or verify dosage requests as being within acceptable ranges. In some circumstances, CDS can be used to calculate dose modifications, for example, using pharmacogenomic-based rules that recognize the effect of particular gene markers on the responsiveness of the patient to a dose, based on body surface area in pediatrics, or as a function of renal status or age; these rules tend to be algorithmic. Dose administration determination may involve more elaborate considerations, as in the case of an insulin sliding scale, or calculation of portals and beam configurations for radiation therapy. These tend to be algorithmic, but may use heuristics and pattern recognition and statistics for optimization.

Prognosis estimation. The CDS question here is to predict the likelihood of good outcomes, morbidity of various types, and mortality. Thus prognostic evaluation of consequences should be an important consideration before treatment selection. Database prediction, as in systems like ARAMIS (Bruce and Fries, 2005), is an ideal approach when the data are available and sufficiently structured. Methods of analysis include logistic regression, classification and regression trees, Bayesian network modeling, and artificial neural networks. Methods for modeling future chance processes, such as Markov modeling and for assessing quality of life, such as the calculation of quality-adjusted life years (QALYs) (Richardson and Manca, 2004) or severity of illness, such as the Apache III score (Kim et al., 2005), are essential underpinnings.

3.2.1.3 Optimizing process flow and workflow

We have described multistep algorithms, guidelines, and protocols in Chapter 2 as a more complex form of CDS. They arise and have use in a variety of settings, because medical care often requires sequences of tasks, with intermediate decision points and pathways. The intent is to help to guide the user in the proper sequence of decisions and actions, to be sure all appropriate alternatives are considered, and to avoid proceeding along inappropriate pathways.

An example is the progression from less expensive and simpler tests to more expensive and invasive procedures in the evaluation of heart disease or breast cancer. Another is the initiation of single-drug therapy for hypertension, with adjustment, substitution, or addition/deletion of medications based on response, side effects, and complications. These are best conveyed to a user by clinical practice guidelines, flow charts, protocols, or flow sheets.

Sometimes multiple tasks, such as orders for tests and procedures, must be done concurrently, but the next step must await at least some of the results from those

tests and procedures before proceeding; and the entire process may involve multiple participants (both human and information-system-based). In these settings, the coordination and communication among participants are important, and a goal is thus to improve workflow, and to maximize speed or efficiency. In this circumstance, augmenting guidelines with workflow management capabilities is desirable (Ciccarese et al., 2005).

In other settings, data may change quite rapidly; for example, in an emergency or intensive care unit setting, and the status of patients needs to be assessed at a glance in order to identify who needs near-term attention versus those that are less critical. CDS in the form of dashboards and alarms/alerts that portray these changing statuses can be helpful in such settings. This is further discussed in Chapter 22.

Last, in clinical trials and in some procedures such as renal dialysis, strict adherence to the steps of a protocol are essential, and a CDS tool can help to ensure that this occurs.

3.2.1.4 Monitoring actions – guarding against errors, providing warnings, alerts, reminders, or feedback about performance

While decision support may provide information when sought, or by overtly asking for it (e.g. by invoking a differential diagnosis tool, a dose calculator, or a clinical guideline), another form of decision support works in the background without overt action of the user, and only interacts with the user when there is a reason to do so. This can occur either in the background of real-time interactive applications between a user and the computer, such as in CPOE, or can be asynchronous and decoupled from user actions (e.g. notification by paging or e-mail about arrival of an abnormal test result).

The computer essentially functions as a guardian or silent partner, monitoring the clinical context and what the user is doing, and either interrupting or contacting the user when situations arise that necessitate this. For example, if a potential order is determined to be hazardous (e.g. a medication interacts dangerously with one that the patient is already receiving, or to which he or she is allergic), a warning can be displayed. If a radiological procedure is being ordered for an indication that is not determined to be appropriate, e.g. ultrasound for a potential appendicitis when a CT would be better in most patients, or if a medication is too expensive, is not on an approved formulary, or a generic medication would be an appropriate substitution, a recommendation for the alternative can be displayed. If a critical abnormal laboratory result is obtained, an alert can be triggered that notifies the physician so that appropriate action can be taken. The passage of time may cause reminders to be generated, as for periodic mammograms in a woman over 50, or for flu shots during the winter season for an elderly patient. If a patient has been in an ICU longer than an expected number of days for his or her diagnosis, the physician can be notified.

In all of these cases, the background alerts, reminders, or feedback may be considered useful or thought to be inappropriate or unjustified. A challenge in providing this kind of decision support is to minimize the situations in which they are

inappropriate, lest the false alarms become annoying or in the worst case, are simply ignored because of their frequency. The conditions for generating messages must be properly defined to be as helpful and pertinent as possible, and the way in which alerts are provided must allow them to be overridden when justified.

3.2.1.5 Focusing attention and enhancing visualization

Another form of CDS is quite indirect and subtle. This is the encouragement of best practices through use of techniques to organize and present information and options in such a way that they serve to enhance recognition of important aspects and remind or facilitate good choices. This may be done either by providing a framework for describing sequences of interactions between the computer and user, such as dialogues that are controlled by an underlying guideline or flowchart, by associative groupings of documentation elements on display screens or printed forms and documents, or by a variety of visualization methods.

With respect to the associative grouping of elements, in particular, if it is done well, it can not only focus attention, but can also offer benefits of improved efficiency, because groups of items can be either selected as a unit when acceptable, or at least brought together for consideration and action decisions rather than each element needing to be sought for by a user and selected or entered individually. Order sets are an example, in that the grouping of orders for a particular indication, such as admission to the cardiac ICU, is pre-established for ease of use by a physician and for ensuring that the physician does not forget to consider including medications for control of pain and anxiety, and for anticoagulation, vital sign and ECG monitoring, diet, cardiac enzyme testing, and other typically considered tasks. If done well, such order sets have the virtue that they not only encourage desirable actions but also, by anticipating likely actions, facilitate workflow and save time.

Besides order sets, another application of this approach to CDS is the design of data entry forms for structured capture of information. Examples might be a form for recording a neonatal visit, an anesthesiology preoperative note, or a specialist referral for cardiac surgery evaluation. The form can include items that are automatically filled in, where possible, from stored data. It can include suggested items that are predetermined to be important, and thus serve as both a handy checklist for recording them and as a reminder to be sure to do so.

Yet another illustration is the generation of reports from structured elements, for example, the printout of a prescription order, or the production of a postoperative note or discharge summary. The design of the report is intended to be easy to automatically generate from stored elements, consistent in appearance, independent of the user who produced it, and well formatted and organized. Such predictability (as well as legibility) facilitates readability and usefulness.

Lastly, we include in this group all forms of visualization that tend to facilitate appreciation of data and their trends, and recognition of situations that may require attention or action. This includes graphics and trend plots,

summarizations, dashboards, and flow sheets. The goal is to support cognitive tasks for managing complexity, usability of systems, and workflow to lead to optimal decision making.

3.2.2 Design of CDS: components and interactions

We now consider the structural aspects of CDS. We do so by identifying a set of functional components of CDS and its invoking environment, and how these components interact. As I noted at the beginning of this chapter, this discussion is somewhat artificial, in that it presents an idealized model of how CDS should be created. What I mean by "idealized" is that, if the design of a CDS capability clearly identifies each of these components, separates them cleanly, and addresses the design of each component in a standardized way, the goals of widespread dissemination and use of CDS can be greatly facilitated. We focus on the idealized model while recognizing that much of CDS is not implemented that way. I maintain that all the components we discuss next that are needed for CDS are present in one form or another in any implementation, but they are not always separable, in terms of the actual software code that implements them. Nonetheless, we can consider these components and the functions they perform individually, at least from a conceptual point of view.

Another idealization I adopt is to refer to a unit of software that provides CDS as a CDS *module*. The heart of a CDS module is a method of transforming input parameters to a patient-specific output. To be modular, the CDS software should be cleanly separable from surrounding or invoking software code, communicating with it via a well-defined interface. Recognizing the many possible implementation methods for CDS that may not be modular at all, we nonetheless use this term to direct our focus to the portion of software directly concerned with the provision of CDS functionality, and to the nature of the interactions by which it relates to the invoking environment.

To provide CDS, several tasks must be performed, as shown in Figure 3.2:

- The CDS module is *initiated* or invoked by some process in the application environment.
- The module *obtains data* through an interface with the application environment, where the data are *entered* by a user, *retrieved* from the EHR or other source, or *provided directly* by the invoking entity. The latter might include context-specific information about the application, user, setting, and function being performed.
- The module *applies knowledge* (e.g. facts in the form of rules, algorithms, or semantic relations), that is either local to the module, or retrieved from a knowledge base.
- A process is then executed that *transforms* the input parameters and knowledge according to the specification of some sort of *decision model* to generate a

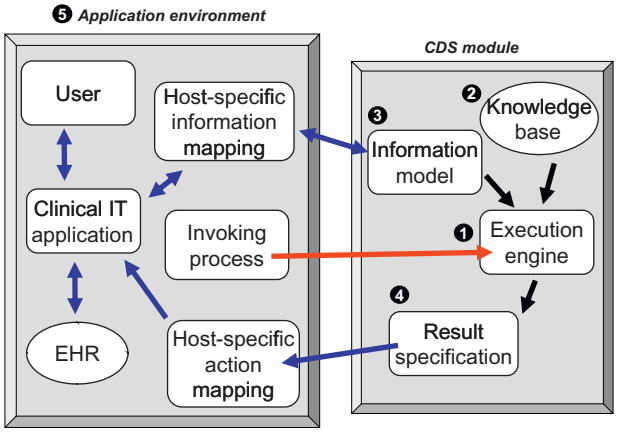

FIGURE 3.2

A conceptual model of CDS design components and their interactions with the host application environment.

patient-specific output. The decision model is usually embodied in an algorithm or computational procedure of some sort.

- The module then *produces a result* that must be communicated to the application environment. That result is usually a recommendation for action.

To carry out this process, the design of a CDS module conceptually has four design elements, or components, and operates in conjunction with an application environment, which is thus considered to be the fifth component (see Figure 3.2). The application environment determines how and when the CDS module gets invoked, how it obtains data and communicates its results, how it interfaces with host software and hardware, and how it interacts with its users. The application environment can be so varied that the specification for this component is only defined with respect to CDS in terms of the nature of the CDS module's interactions with it.

3.2.2.1 Decision model/execution engine

All kinds of CDS have some kind of execution paradigm, that is, a method of organizing or processing input information, to produce some kind of output, or result. The sequence in which data are requested and the algorithm or method for processing data depend on an underlying model of the decision problem. For example, an alert or reminder may be designed to be triggered by an event, such as a mouse click, the arrival of a lab result, or the passage of time, to obtain specific data items. The process then evaluates a Boolean logical condition expression about the data, to determine the truth value for the expression. If the Boolean condition

evaluates to "true," the alert or reminder may then cause an e-mail, page, or displayed message to be generated, in order to notify an appropriate physician. The *decision model* in this is *Boolean expression evaluation*.

A differential diagnosis program may collect data and evaluate the diagnostic possibilities using Bayes' theorem: the Bayes' theorem algorithm is the underlying decision model. A dose therapy calculation tool might use a formula that needs such parameters as body surface area, renal status, or age to make recommendations for modifications of medication dose for children, those with kidney failure or the elderly; the decision model is a computational formula.

The decision models that are used in CDS rely principally on the methods discussed in Chapter 2. These include:

1. Information retrieval; that is, the model by which data and knowledge are used to select pertinent items to retrieve
2. Logical expression evaluation
3. Probabilistic and data-driven classification/prediction
4. Heuristic methods and expert systems
5. Calculations, algorithms, and multistep processes
6. Associative grouping of elements; that is, the model determining what these associations are and under what conditions they are activated.

Conceptually, we can consider that the decision model, to the extent that it involves computational processes, is embodied in an *execution engine.* The execution engine is the part of the CDS software that evaluates data to produce output. As we have noted, actual implementations may not cleanly separate this code from other parts of an application, or even from other parts of the CDS module itself, but advantages can be realized if that can be done. Principally, this separation allows the execution engine to be refined and enhanced as improvements in the way it should operate become understood. Also, this provides flexibility and portability, in that the execution engine can be recoded and reimplemented in different platforms independently of other CDS parts, and can even be embodied in external services (see Chapter 29).

3.2.2.2 Knowledge content resources

Sometimes, as in an application for recommending electrolyte replacements in acid-base disorders, the calculations and sequences of actions are embedded in software code. However, as we noted earlier, it is often helpful to implement the general methodology of a particular decision model as an execution engine that can be used to apply the method to all knowledge of that type. For example, if the electrolyte replacement algorithm can be represented in a flow chart modeling formalism, then a guideline execution engine can run it. To be most flexible, ideally, the knowledge – the formulas or equations, the logic of production rules, the flow charts, and so on – should exist external to the "engine" that accepts inputs of that type, processes the knowledge, and produces a result. Separating the knowledge from the

engine, when it is possible to do so, enables the engine to operate on a variety of similar kinds of knowledge.

- As a consequence, the knowledge resources can be managed independently. For example, they can be authored and edited through use of a knowledge editor tool.
- With appropriate editor functionality, an editor tool can display the knowledge in a form that is readable by a human subject expert rather than requiring the skills of a software engineer or other technical support person.
- If the knowledge is made transparent in this fashion, maintenance, review, and update are easier to do.
- If a standard format is used for encoding the knowledge, or for importing and exporting it from external repositories, the knowledge can be shared and disseminated.
- If the decision model evolves, e.g. in terms of the ability to use more refined or detailed knowledge, the knowledge base can be updated separately to incorporate those knowledge elements.

Knowledge content resources can be structured or unstructured, depending on the purpose and the computational requirements of the decision model. For example, if the purpose is simply to retrieve and display information in human-readable form, the only structure required might be the use of index terms or keywords, to facilitate retrieval by a search engine, although with no structure at all, a text-based search such as by Google® is still possible.

If the knowledge is a logical expression for a production rule, then the expression must obey the syntactic conventions needed to evaluate the expression in whatever language or formalism is used. In MYCIN, one of the earliest rule-based systems in medicine, production rules had the format IF *condition* THEN *action*, where *condition* was a Boolean logical expression with "certainty factors" associated with the terms. The execution engine could evaluate the conditional expression and had an algebra for combining the certainty factors to produce an updated certainty factor associated with the assertion in the *action* part of the rule (Shortliffe, 1976), and could control the sequence of execution of rules through a goal-driven, backward chaining heuristic. In alerts and reminders in Arden Syntax (Hripcsak, 1991), an *evoke* section defines triggering event(s), a *data* section specifies the data elements used, a *logic* section defines the procedure to evaluate the data elements in a Pascal-like syntax, and an *action* section defines the task to be carried out if the logic section evaluates to *true*. An Arden Syntax interpreter or compiler could then serve as an execution engine to process a knowledge base of Arden Syntax rules, evaluating a rule when triggered by appropriate evoking conditions. GELLO (Sordo et al., 2004), a more recent HL7-approved ANSI-endorsed standard expression language supporting HL7's version 3 Reference Information Model (RIM) (RIM, 2006), defined an object-oriented syntax for specifying logical conditions. Knowledge bases encoded in GELLO expressions could be evaluated by a GELLO execution engine. Still more recently, in 2013, the Health eDecisions

initiative sponsored by the US Office of the National Coordinator for Health IT proposed a model-based framework for exchange of computable knowledge artifacts, which has been approved as an HL7 standard (HL7_HeD, 2013). This specifies a model based on ontologies for identifying events, conditional expression syntax and clauses, and actions, and for specifying data and knowledge elements needed in CDS rules, order sets, and documentation templates. Based on this formal modeling process, knowledge can be exported in XML or can, if adapters are written, be converted into any other model formalism and its associated expression language.

The knowledge base for a Bayesian diagnosis tool would be the prior probability distribution for the diseases to be considered, and the conditional probabilities of findings for each of the diseases (Warner et al., 1964; Lodwick, 1965; deDombal, 1975).

A guideline interpretation engine that is designed to support traversal of a guideline and interactive acquisition of data and evaluation of conditions to determine next steps could operate on guidelines encoded in a knowledge base according to a formalism that the interpretation engine understands. Examples of this are the guideline engines supporting representation formats known as Pro*forma*, GLIF, EON, Asbru, and SAGE, as discussed in Chapter 2 and reviewed further in Chapter 16.

3.2.2.3 Information model

CDS requires a precise specification of the kinds of information the computation model will utilize, which we refer to as its *information model*. The knowledge content resources typically contain statements, facts, conditional expressions, or other relations that refer to or operate on patient-specific data. If we formally specify the information model, this provides flexibility in that the same CDS resource can be used in more than one kind of setting; for example, interactively with a user as well as in background mode, retrieving data from the EHR, and in more than one platform and system environment. The specification must include not only the *format* of the data that the CDS module receives and uses but the *taxonomy* or coding scheme for its labels and also for any of its coded/categorical values, and the restrictions on value sets that are allowable. For example, if one is seeking to run a rule about medication interactions, it is important to know that they are encoded in RxNorm (NLM, 2005), or that a diagnosis is encoded in SNOMED-CT (SNOMED, 2006), or that a lab test is encoded in LOINC (Regenstrief, 2005). Value set restrictions may confine the medications of interest to angiotensin converting enzyme inhibitors, or diagnoses to cardiovascular diagnoses, or lab tests to particular enumerated tests. The specification should also involve *grounding* the data elements in terms of precise attributes like units, method of obtaining them, time frame, etc. This is discussed further in Chapters 17 and 18, in terms of documentation elements or *archetypes*.

Note that beyond defining these requirements, the adaptations necessary for accessing or obtaining them are not the province of the CDS module but of the

application environment. If the data are to be obtained by interaction with a user, the host may also need to include an external display name for a data entry field. If the input that is allowed needs to be validated (e.g. checked for limits of a numeric range, length of a text string, the presence of valid characters, or conformance with items on a predefined pick list or dictionary), then those criteria for validation (and the content of the pick list or dictionary) need to be adapted from the information model or added by the application environment.

If the data are to be retrieved from a stored repository, then either the information model used in the host application environment should be the same, or a process for mapping the data elements from it needs to be established. Some of the data elements may need to be transformed, if differences in the definitions of those elements in the host EHR and in the CDS module's information model require it. For portability of CDS, the use of a standardized information model such as the HL7 v.3 RIM, for example, should be used to evaluate a GELLO logic expression. In general, it is unlikely that real operational clinical information systems will use a standard reference information model (such as the HL7 RIM or some other) directly in its implementation. Thus for interoperability between systems, or to use externally developed CDS, a mapping would need to be developed for each implemented vendor-specific system between that standard reference information model and the vendor system information model.

Arden Syntax uses a different approach, in which each Arden Syntax rule's data section allows customization to indicate how the various data elements should be retrieved from a particular host information system. This information is encoded within curly braces, indicating that it is not part of the general rule, but is host-system-specific. The disadvantage of the Arden Syntax approach is that each rule must be customized individually for each host system. Also, even if multiple rules all refer to similar data items, each rule must include the customization statements for each of the data items.

3.2.2.4 Result specification

Operation of the CDS execution engine is carried out with the goal of producing some output, whether in the form of retrieved resources, a calculated result, a recommended action, or a data entry screen or formatted report. Since the result is dynamically determined through execution, there needs to be an explicit process for determining how that output gets produced.

The result specification could be regarded as part of an expanded view of the information model, but we consider it separately because of its distinct role in the CDS process. Note that the result of a CDS process can be a set of one or more values of clinical parameters, but it can alternatively be an action (possibly including clinical values). Actions are verbs like notify, schedule, order, cancel, etc.

For example, the result of evaluating an Arden Syntax rule to *true* might be an action to be performed, such as to send a specific message to the attending physician. A calculation of dose of a medication based on adjustments for renal function or age might produce a result in terms of a modified recommended dose.

A Bayesian differential diagnosis program's result would be the set of diagnoses with their posterior probabilities. Traversal of a clinical practice guideline algorithm based on evaluation of entered or retrieved patient-specific data values would produce a result at each step, indicating the optimal next step.

Thus conveying the result to an application environment involves mapping the result to the performance of actions or production of outputs that the application expects to carry out. This will largely be the responsibility of the application environment, as discussed in the next section. What is needed in the CDS module is a taxonomy of kinds of results that can be produced from decision support, to facilitate such host mappings. Some early work in the execution of clinical guidelines (Essaihi et al., 2003; Tu et al., 2004) produced such a taxonomy. This has also been pursued further in a CDS Taxonomy produced by the National Quality Forum Committee on CDS (NQF, 2010), and in the action ontology of the Health eDecision knowledge model (HL7_HeD, 2013).

Modularity of design is one of the reasons that we consider the result specification in CDS separately from the mode of interaction with the user or applications in the application environment. Just as with the specification of the information model for data used in the decision model, separation of the result specification enables a decision support capability to be adapted to several possible modes of interaction in any of a variety of applications on different platforms. For example, the result could be provided in real-time in interactive applications, in the background in alert/reminder usages, or in batch mode in the production of reports or summaries. If the result is a set of one or more values, this might call for the EHR to be updated with that information. If the result is an action to be performed, this needs to be interpreted by the host environment in terms of its ways of performing that action. For example, an action may call for a physician to be notified about an abnormal value, which could be done by e-mailing the physician, displaying an interruptive popup message on a screen, transmitting the recommendation to another application, or storing it in a pending task list to be seen when the physician next logs into the system.

3.2.2.5 Application environment

As we have seen, many of the features of CDS operation are not embodied in the CDS module itself but in the application environment that invokes it. The application environment determines how the CDS module communicates with a user, such as via an interactive dialogue, or obtains data from the EHR or other sources, and how it conveys its results. The application environment can also pass to the CDS module certain context data such as those describing the application setting, the user, and the purpose.

The degree of integration of CDS with applications is one of the most critical factors for determining its success, yet too tight an integration limits the ability to achieve portability and reuse of CDS modules. We will begin the consideration of the nature of the interaction of a CDS module with the application environment by revisiting a simple example we used in Chapter 2 to illustrate the range of options to consider and the complexities involved – even for a simple form of CDS.

The example relates to the set of rules regarding the handling of abnormal and critical laboratory test result values; in other words, results that exceed predefined limits and require flagging or, for critical results, urgent attention. The knowledge regarding such abnormal values can be found in the literature, and the simplest form of decision support is the ability to retrieve references to such abnormal values. This could be in the form of bibliographic citations or Web sites displaying laboratory values and their accepted normal ranges and critical values. Ideally, those latter sites should also cite references to the literature about how to interpret them.

The least integrated way to make this information available would be to enable the user to access the Web and to do a search for it, using his or her own search terms. Slightly more integrated access would be a resources page, which would have a set of predefined links to useful reference information that could be accessed from the clinical IT system. To be of greater value, it would be useful to have access to this information at the point at which a physician is reviewing laboratory results for a patient. The difference between looking up lab result values to determine whether they are abnormal and having a direct link to a particular citation giving that information in the context in which it is needed – that is, when the lab result is being reviewed – is that in the latter case the information needed is preselected and automatically available to the user.

A more useful way to provide this information, which is done in most clinical systems, is to automatically flag the abnormal lab result on the report or display screen that is reviewed by the provider, by a symbol indicating that it is outside of normal ranges. This could then be combined with a link to the available citations to give further information. The flag indicating abnormality could be introduced by the clinical laboratory information subsystem or by the laboratory results report display application, using a formal logical condition expression that is evaluated by the computer to determine the presence of abnormality. Thus it could occur at any of three points; either at the time the result is produced in the laboratory subsystem, when it is entered into the EHR, or when it is displayed.

Another way to deliver an abnormal result finding is by generating an alert message that is sent to the provider, perhaps by e-mail, text page, or interruptive alert. This requires integration into a clinical information system in such a way that information about the particular patient and the appropriate provider are able to be identified automatically. This would also allow more elaborate decision rules to determine whether the result is new or a repeat of an already abnormal value about which the physician has previously been notified, or if there are coexisting conditions that might explain the result (e.g. renal failure).

To be maximally useful, knowledge in the form of a decision rule such as that used for responding to abnormal laboratory test results would exist in a rules knowledge base and be triggered by a variety of different possible event scenarios – for example, the entry of an abnormal lab result into the patient's clinical record, a medication order interaction check with respect to existing lab results, or the flagging of abnormal laboratory results on review by a physician. With the knowledge in a knowledge base, different events such as result entry, order placement,

or display of results could trigger evaluation of the rules indexed according to the various parameters of interest, to determine which, if any, might apply, and then to carry out actions that are appropriate based on the triggering application and depending on the result of the evaluation. For example, in an alerting application, evaluation of the rule to true would result in the generation of a warning message to the provider by e-mail, page, or other means of communication. In an interactive CPOE application, evaluation to true might generate a recommendation in the form of an immediate interactive popup or inline message to decrease the dose of a medication being prescribed. In the result display application, evaluation of truth would result in the flag symbol indicating abnormal result being appended to the value.

The first usage described, that of displaying a citation, simply requires that the knowledge about abnormal lab results be available in text form in some defined location (e.g. in a bibliographic database). It requires no computability, just the ability to retrieve it.

The second usage, access via a direct link from the result information display, can be done by manually identifying the appropriate citation to be displayed whenever abnormal results occur. To be more useful, however, retrieval based on the context, for example, could work as follows: By recognizing that the context is a laboratory results display application, the CDS tool could determine that information pertaining to abnormal laboratory results would be useful, and a specialized retrieval program could be designed to select the kind of information to be retrieved from a general retrieval search engine by passing context-specific parameters related to clinical laboratory abnormal results. This constitutes a kind of "information broker" function, and is exemplified by the infobutton manager described in Chapter 2 and further elaborated in Chapter 19.

The third usage – that of flagging abnormal results – requires the presence of a formal computable expression that can be evaluated by the computer. This would be of a logical format such as "if lab test result *y* exceeds threshold *a* then return true." For this to work, the value of the lab test result of interest must be assigned to the parameter *y* and the upper normal range for the lab test result must be assigned to the parameter *a*. The software application must also know that if the result True is returned from the evaluation, then a flag value such as * or # should be appended to the display of the laboratory result. Thus this application usage requires a formal expression, an evaluation engine, and a simple set of data parameters that can be passed to the evaluation engine and returned from it.

The fourth usage is the execution of a generalized rules interpreter in the context of an event-driven architecture, in which particular rules are evaluated as a result of triggering events, and the actions performed depending on the result of the evaluation are a function of the application that generated the trigger. This approach not only requires a knowledge base, an indexing scheme for accessing the rules in the knowledge base, and an execution engine, but it requires also a means of integration with various possible invoking applications, as well as, for example, in the case of notifications to physicians, the ability to invoke other applications. This usage has maximum flexibility and power, because the same piece of knowledge – in this

case, any logical expression regarding what constitutes an abnormal laboratory test result value – can be used in a variety of different contexts. Thus the decision rule itself only needs to be developed once, and can be maintained or updated if necessary in one place, and if properly set up, all the applications that utilize it can be identified, should there be a reason to change the rule in the knowledge base.

We can consider a variety of other kinds of clinical decision support usages that range from passive to active and from loose to tight integration with applications, and do a similar decomposition of the necessary elements. Work needs to be done to define the extent to which application-specific behavior can be further abstracted into a taxonomy of result types, as discussed earlier, so that more of the functionality can be moved into CDS modules rather than requiring custom interfacing and handling of results in the application environment.

3.2.3 Modes of interactions

One of the challenges in providing CDS is that of determining the most effective way to interact with humans, so that the advice is as patient-specific and timely as possible. At the same time, it must be acceptable to the human user, by not requiring a lot of extra work, being disruptive to workflow, or being redundant. Also, given the role of decision *support* rather than decision *making*, advice must be given in a fashion that recognizes human decision making prerogatives and avoids being inflexible or insistent when it is not necessary.

Figure 3.3 lists a set of axes that can serve as a guide for thinking about the various dimensions involved in providing CDS and interacting with users, which determine the extent and manner of integration of CDS with the clinical IT application environment. We consider each of these briefly.

3.2.3.1 Locus of control

A CDS instance can be *initiated* by a user when the need for help is recognized; for example, in seeking to find an answer to a question, or obtaining assistance in assessing a diagnostic or therapeutic decision. Or it can be initiated by the computer, such as in processes aimed at monitoring user actions to guard against errors or detecting suboptimal practices; for example, in detecting an inappropriate order, or a time interval at which a mammogram should be ordered or HbA1c level should be checked. The *locus of control* thus either resides with the user or the computer in these examples. There are also intermediate situations, in which CDS resources are made available and have the means to automatically be context-aware, as in infobuttons or patient-specific guidelines, but it is up to the user to initiate their use.

3.2.3.2 Degree of assertiveness

Decision support capabilities can be provided to a user with varying degrees of insistence or "assertiveness." This only applies, of course, to settings in which the computer is the locus of control and initiates the CDS instance. The most passive way in which decision support can be offered would be to simply present a discussion of a topic that can be read by the user. One step up from this would be

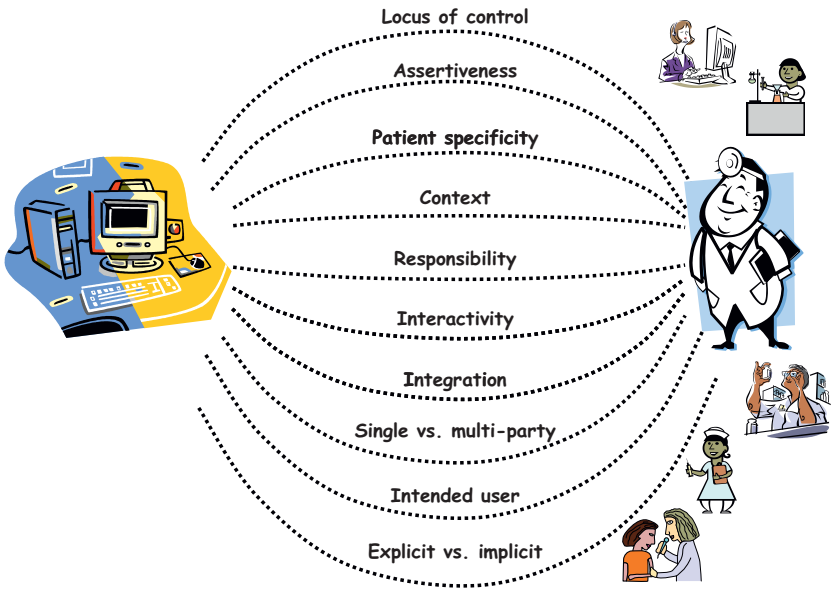

FIGURE 3.3

Dimensions of computer-user interaction in CDS.

to present specific recommendations and advice, although again in a form that is simply to be read. A slightly more insistent form of decision support would be the requirement that the information provided be acknowledged. A further increase in assertiveness might involve a list of choices of action; for example possible medication regimens, allowing "other" as an additional option or an override of the recommended dose of a medication if a justification is provided. Still more assertive would be forced choice among alternatives without the possibility of override. The most active form of decision support would be a closed loop process in which actions occur automatically in response to inputs, although these could be inspected or monitored by a user. Implanted cardiac pacemakers use this mode of operation, but for reasons we have noted previously, there are very few other instances of such closed-loop processes in routine use.

3.2.3.3 Patient specificity

Comprehensive knowledge base resources, such as the medical literature in PubMed, a guideline repository, or a collection of possible alerts and reminders, are important to the ability to provide robust decision support. But a challenge in delivering effective decision support is to be able to select resources that are relevant for a given patient and to determine when and where, or even if, they should be used in a particular setting.

To accomplish this, the CDS module needs to be able to obtain information about the clinical problems or findings, care setting (e.g. office visit, phone call, or

hospitalization) and other patient-specific parameters, as specified by the logic of the rules under consideration. This involves the ability to access the EHR or obtain data from the user or from the invoking application.

3.2.3.4 Context

Beside the patient characteristics and setting of the clinical encounter, context affects the nature of decision support in several ways. The principal contextual factor is the kind of application and the function being performed when CDS is invoked. For example, if the context is an interactive CPOE program, decision support would likely need to be very responsive so as not to perceptibly delay or impede the real-time interactivity. If the context is a background process aimed at collecting data on and evaluating conformity with care pathways in various units of a hospital (e.g. the coronary care unit or the postoperative orthopedic floor), it might be suitable to communicate deviations from expected targets to providers at the beginning of the workday or at rounds. Event-driven panic alerts indicating critical abnormal laboratory values would need to be communicated to providers by any available means as quickly as possible and require acknowledgment. Preventive screening or immunization reminders to a patient might have a response window of a week or even a month.

The same knowledge can be used in different contexts and practice settings. To return to an example discussed earlier, consider a rule that performs an action if a lab result exceeds a threshold: this might be used in an alert that is automatically triggered when a result is produced by the laboratory, or the rule might be used during the generation of a lab test result review display screen, for flagging abnormal results. The multiplicity of uses and contexts for decision support knowledge is one of the rationales for the need to construct knowledge bases containing collections of decision support content. The content in a knowledge base can be indexed or meta-tags provided pertaining to various axes, such as those relating to context, setting, medical problem, and CDS purpose, that enable specific knowledge to be retrieved and used when needed. Another consequence of this capability is that it would be necessary for only one instance of a particular item of core knowledge (e.g. a decision rule,) to exist, making it easier to maintain the knowledge, review it, update it, and propagate changes to all the applications that use variations of it.

3.2.3.5 Interactivity

Somewhat related to the classification of decision support in terms of degree of assertiveness, patient specificity, and context is the degree of interactivity. Knowledge resources may be in the form of static human-readable information, such as text or a table, retrieved in response to specific search. Such a presentation could be viewed and examined but require no specific action by the user. An interactive mode of delivery of this same information would be one for which some entry or acknowledgment is required by the user.

A more interactive form of CDS might be a computational tool that produces a similar kind of text or tabular report, but which provides the ability to manipulate parameters that are entered into it, as, for example, a computation tool for drug

dose calculation based on body surface area, or a tool for performing sensitivity analysis of a decision analysis model to determine how stable its decision is as a function of change in the estimate of prior probability of disease or the riskiness of a treatment. Often, interactivity occurs in CDS in the form of requests for data to be entered or selection of options by the user.

Still other aspects relating to interactivity are illustrated by the various ways in which recommendations can be communicated to the user and the options available to the user for responding to or overriding recommendations. Some decision support messages might be provided in the form of noncritical alerts or reminders, generated in the background and only seen when a user next logs into the information system and requiring no further action on the user's part. Alternatively, for more important alerts or reminders, they might be sent by e-mail or pager. Various graphical renditions, dashboards, or summaries might be available as options for viewing complex data and relationships.

3.2.3.6 Degree of integration with clinical IT applications

A CDS capability can be integrated into the clinical information system to greater or lesser degree. CPOE is an example of an application with a need for a high degree of integration of CDS. A drug interaction checking tool would be most useful during CPOE if it were integrated with another system that maintained a list of current medications for the patient. Given the EHR, a CDS tool could also include automatic checks against allergies or other patient-specific contraindications.

3.2.3.7 Single vs. multi-party focus

Some applications of decision support, such as computer-based clinical practice guidelines, have the potential for optimizing the care process by suggesting appropriate next steps. In a busy practice environment or inpatient setting, automation of guidelines could also help to optimize workflow by coordinating scheduling and use of resources and activities of participants through communication and synchronization functions (e.g. don't do task B until task A is completed), and monitoring the times, delays, and statuses of expected events. Note that the parties involved may be human or computer-based (e.g. a scheduling system or messaging system).

Notification of alerts is another example of an application that may have a multi-party aspect. Typically, if a critical lab result needs to be acted on by someone, there is a set of processes defined for notifying a patient's primary physician about such an alert (e.g. some sequence of page, telephone message, e-mail, or text message, with requirement for acknowledgment), and a defined sequence for notification of other providers if that person does not respond within a specified period of time.

3.2.3.8 Intended user

Decision support for various purposes may be designed for different kinds of intended users; for example, direct support of physician decision making; aids to nurses, pharmacists, laboratory or radiology technologists, emergency medical

technicians (EMTs) or paramedics; reports of utilization of resources, errors, or costs to managers; or information resources and decision aids for patients and the general public. The kind of knowledge involved, the decision-making approach used, and the mode of operation may vary considerably depending on user and purpose.

3.2.3.9 Explicit vs. indirect support

Calculation tools, guidelines, alerts, and reminders all are designed to give specific advice or recommendations. But the other kind of decision support we have included is one in which the organization, grouping, sequencing, or mode of visualization of information presentation is intended to foster optimal decision making in a more subtle manner, simply by focusing attention, serving to remind the user of items that might otherwise be forgotten, encouraging systematic consideration and possibly influencing prioritization. We mention again some examples of this mode of decision support, namely the use of structured data entry forms, order sets, templates for reports and summaries, dashboards, flow sheets, and protocols.

3.3 Other considerations

We have touched on a number of settings and contexts in which decision support could be used, but which will not be discussed in detail in this book. One of the primary other uses is in the realm of education and training. Not only can decision support knowledge bases be useful as educational reference tools, but the decision support can be used directly in a dynamic way in case-based problem-solving exercises – simulations of clinical problems requiring intervention by a user and feedback about the appropriateness of the actions taken. Methods analogous to CDS may be used to generate a range of variation of clinical parameters in a simulation, with the inclusion of a random component as well. CDS-like capabilities can be used in a critiquing mode, in which actions are performed by the user first and then evaluated by the CDS-like resource for conformance with the underlying decision model (e.g. a guideline). Or decision support may be used in what is known as an "intelligent tutoring system" mode of operation, to probe student responses or actions in terms of their similarity to prototypical problems in such situations, and to tease out the underlying misconceptions.

We will not delve further into image and signal processing, pattern recognition, and feature extraction, as these are largely embedded in niche applications, and our focus is on more generic CDS capabilities and the issues of deploying them in a health care enterprise.

Much of the development of CDS to date has been something of an art form, with creative individuals identifying innovative and useful ways of providing it and showing effectiveness. Because of a lack of well-defined principles, the discovery process often has had to be replicated by others, sometimes with painful and

disappointing results. The collective body of experience in the literature is nonetheless quite large.

Although there is new impetus to moving ahead, the lessons of the past need to be recognized if we are not to be destined to repeat the mistakes that have limited progress over the past 50 years or more. A goal of this book is to begin to move toward a formal understanding of the requirements for CDS, based on the lessons and experiences of the past, clarifying an understanding of the requirements for infrastructure, standards, and business/organizational strategies that will lead to success.

The task of providing and maintaining robust CDS capabilities is a long and complex undertaking. It is important not to oversimplify it, or to rush to deploy CDS without adequate preparation, lest unsatisfactory results occur, bad press be generated, and an era of discouragement take hold. We seek to increase awareness and understanding of what the effort requires and to begin a systematic approach to tackling the problems that have vexed the field and held it back over these many years.

References

Abendroth, T.W., Greenes, R.A., 1989. Computer presentation of clinical algorithms. MD Comput. 6 (5), 295–299.

Altenstetter, C., 2003. EU and member state medical devices regulation. Int. J. Technol. Assess. Health. Care. 19 (1), 228–248.

Barnett, G.O., Cimino, J.J., Hupp, J.A., Hoffer, E.P., 1987. DXplain. An evolving diagnostic decision-support system. JAMA 258 (1), 67–74.

Berlin, A., Sorani, M., Sim, I., 2006. A taxonomic description of computer-based clinical decision support systems. J. Biomed. Inform. (Epub ahead of print).

Berner, E.S. (Ed.), 2010. Health Inform. Springer

Bindels, R., Hasman, A., Kester, A.D., Talmon, J.L., De Clercq, P.A., Winkens, R.A., 2003. The efficacy of an automated feedback system for general practitioners. Inform. Prim. Care. 11 (2), 69–74.

Bodenheimer, T., 2003. Interventions to improve chronic illness care: evaluating their effectiveness. Dis. Manag. 6 (2), 63–71.

Bruce, B., Fries, J.F., 2005. The Arthritis, Rheumatism and Aging Medical Information System (ARAMIS): still young at 30 years. Clin. Exp. Rheumatol. 23 (5 Suppl. 39), S163–167.

Christensen, C.M., Raynor, M.E., 2003. The Innovator's Solution. Harvard Business School Press, Cambridge, MA.

Ciccarese, P., Caffi, E., Quaglini, S., Stefanelli, M., 2005. Architectures and tools for innovative health information systems: the guide project. Int. J. Med. Inform. 74 (7–8), 553–562.

Cimino, J.J., Li, J., Bakken, S., Patel, V.L., 2002. Theoretical, empirical and practical approaches to resolving the unmet information needs of clinical information system users. Proc. AMIA. Symp. 170–174.

Costantini, O., Papp, K.K., Como, J., Aucott, J., Carlson, M.D., Aron, D.C., 1999. Attitudes of faculty, housestaff, and medical students toward clinical practice guidelines. Acad. Med. 74 (10), 1138–1143.

deDombal, F.T., 1975. Computer-aided diagnosis and decision-making in the acute abdomen. J. R. Coll. Physicians. Lond. 9 (3), 211–218.

Del Fiol, G., Curtis, C., Cimino, J.J., Iskander, A., Kalluri, A.S., Jing, X., et al., 2013. Disseminating context-specific access to online knowledge resources within electronic health record systems. Stud. Health. Technol. Inform. 192, 672–676.

Del Fiol, G., Huser, V., Strasberg, H.R., Maviglia, S.M., Curtis, C., Cimino, J.J., 2012. Implementations of the HL7 context-aware knowledge retrieval ("Infobutton") Standard: challenges, strengths, limitations, and uptake. J. Biomed. Inform. 45 (4), 726–735.

Embi, P.J., Leonard, A.C., 2012. Evaluating alert fatigue over time to EHR-based clinical trial alerts: findings from a randomized controlled study. J. Am. Med. Inform. Assoc. 19 (e1), e145–148.

Essaihi, A., Michel, G., Shiffman, R.N., 2003. Comprehensive categorization of guideline recommendations: creating an action palette for implementers. AMIA. Annu. Symp. Proc. 220–224.

Feigenbaum, E.A., Feldman, J. (Eds.), 1963. Computers and Thought. McGraw-Hill, New York.

Freedman, A.N., Seminara, D., Gail, M.H., Hartge, P., Colditz, G.A., Ballard-Barbash, R., et al., 2005. Cancer risk prediction models: a workshop on development, evaluation, and application. J. Natl. Cancer Inst. 97 (10), 715–723.

Giger, M.L., 2004. Computerized analysis of images in the detection and diagnosis of breast cancer. Semin. Ultrasound. CT. MR. 25 (5), 411–418.

Greenhalgh, J., Long, A.F., Flynn, R., 2005. The use of patient reported outcome measures in routine clinical practice: lack of impact or lack of theory? Soc. Sci. Med. 60 (4), 833–843.

Hackett, J.L., Gutman, S.I., 2005. Introduction to the Food and Drug Administration (FDA) regulatory process. J. Proteome. Res. 4 (4), 1110–1113.

Harding, J., 1994. Practice guidelines. Cookbook medicine. Physician. Exec. 20 (8), 3–6.

Harpole, L.H., Khorasani, R., Fiskio, J., Kuperman, G.J., Bates, D.W., 1997. Automated evidence-based critiquing of orders for abdominal radiographs: impact on utilization and appropriateness. J. Am. Med. Inform. Assoc. 4 (6), 511–521.

HL7_HeD, 2013. HL7 Implementation Guide: Clinical Decision Support Knowledge Artifact Implementation Guide, Release 1.

Hripcsak, G., 1991. Arden syntax for medical logic modules. MD Comput. 8 (2), 76, 78.

IBM Research, 2013. The DeepQA Research Team Retrieved October 21, 2013, from: <http://researcher.ibm.com/researcher/view_project.php?id=2099>.

Jiang, Y., Nishikawa, R.M., Schmidt, R.A., Metz, C.E., Giger, M.L., Doi, K., 1999. Improving breast cancer diagnosis with computer-aided diagnosis. Acad. Radiol. 6 (1), 22–33.

Johnson, H., 1994. Relationship between user models in HCI and AI. Comput. Digit. Techn. IEE Proc. 141 (2), 99–103.

Jonnalagadda, S.R., Del Fiol, G., Medlin, R., Weir, C., Fiszman, M., Mostafa, J., et al., 2013. Automatically extracting sentences from Medline citations to support clinicians' information needs. J. Am. Med. Inform. Assoc. 20 (5), 995–1000.

Kawamoto, K., Houlihan, C.A., Balas, E.A., Lobach, D.F., 2005. Improving clinical practice using clinical decision support systems: a systematic review of trials to identify features critical to success. BMJ 330 (7494), 765.

Kawazoe, Y., Miyo, K., Kurahashi, I., Sakurai, R., Ohe, K., 2013. Prediction-based Threshold for Medication Alert. Stud. Health. Technol. Inform. 192, 229–233.

Kim, E.K., Kwon, Y.D., Hwang, J.H., 2005. Comparing the performance of three severity scoring systems for ICU patients: APACHE III, SAPS II, MPM II. J. Prev. Med. Pub. Health 38 (3), 276–282.

Kohane, I.S., Masys, D.R., Altman, R.B., 2006. The Incidentalome: A threat to genomic medicine. JAMA 296 (2), 212–215.

Kuperman, G.J., Teich, J.M., Gandhi, T.K., Bates, D.W., 2001. Patient safety and computerized medication ordering at Brigham and Women's Hospital. Jt. Comm. J. Qual. Improv. 27 (10), 509–521.

Lee, E.K., Mejia, A.F., Senior, T., Jose, J., 2010. Improving patient safety through medical alert management: an automated decision tool to reduce alert fatigue. AMIA. Annu. Symp. Proc. 2010, 417–421.

Liang, M.H., 1992. From America: cookbook medicine or food for thought: practice guidelines development in the USA. Ann. Rheum. Dis. 51 (11), 1257–1258.

Lodwick, G.S., 1965. A probabilistic approach to the diagnosis of bone tumors. Radiol. Clin. North. Am. 3 (3), 487–497.

Miller, P.L., 1983. Critiquing anesthetic management: the 'ATTENDING' computer system. Anesthesiology 58 (4), 362–369.

Miller, P.L., Black, H.R., 1984. Medical plan-analysis by computer: critiquing the pharmacologic management of essential hypertension. Comput. Biomed. Res. 17 (1), 38–54.

Mugford, M., Banfield, P., O'Hanlon, M., 1991. Effects of feedback of information on clinical practice: a review. BMJ 303 (6799), 398–402.

NHLBI, 2004. The Seventh Report of the Joint National Committee on Prevention, Detection, Evaluation, and Treatment of High Blood Pressure (JNC 7), National Heart Lung and Blood Institute, NIH, U.S. Department of Health and Human Services.

NLM, 2011. RxNorm, UMLS. National Library of Medicine, National Institutes of Health (NIH). Available from: <http://www.nlm.nih.gov/research/umls/rxnorm/>.

NQF, 2010. Driving Quality and Performance Measurement – A Foundation for Clinical Decision Support: A Consensus Report. National Quality Forum (NQF). Available from: <http://www.qualityforum.org/Publications/2010/12/Driving_Quality_and_Performance_Measurement_-_A_Foundation_for_Clinical_Decision_Support.aspx>.

Ogden, J., Fuks, K., Gardner, M., Johnson, S., McLean, M., Martin, P., et al., 2002. Doctors expressions of uncertainty and patient confidence. Patient. Educ. Couns. 48 (2), 171–176.

Regenstrief, 2005, Logical Observation Identifiers Names and Codes (LOINC®). Regenstrief Institute website. Available from: <http://www.regenstrief.org/loinc/>.

Richardson, G., Manca, A., 2004. Calculation of quality adjusted life years in the published literature: a review of methodology and transparency. Health. Econ. 13 (12), 1203–1210.

RIM, 2006. The Reference Information Model (RIM) Health Level Seven, Inc., from: <http://www.hl7.org/Library/data-model/RIM/modelpage_mem.htm>.

Rockhill, B., Spiegelman, D., Byrne, C., Hunter, D.J., Colditz, G.A., 2001. Validation of the Gail et al. model of breast cancer risk prediction and implications for chemoprevention. J. Natl. Cancer Inst. 93 (5), 358–366.

Shiffman, R.N., Greenes, R.A., 1991. Use of augmented decision tables to convert probabilistic data into clinical algorithms for the diagnosis of appendicitis. Proc. Annu. Symp. Comput. Appl. Med. Care. 686–690.

Shortliffe, E., 1976. Computer-based Medical Consultations: MYCIN. Elsevier, New York.

Sim, I., Berlin, A., 2003. A framework for classifying decision support systems. AMIA. Annu. Symp. Proc. 599–603.

Smith, R.A., Cokkinides, V., Eyre, H.J., 2004. American Cancer Society guidelines for the early detection of cancer, 2004. CA. Cancer. J. Clin. 54 (1), 41–52.

SNOMED, 2006. SNOMED CT, SNOMED International. College of American Pathologists.

Sordo, M., Boxwala, A.A., Ogunyemi, O., Greenes, R.A., 2004. Description and status update on GELLO: a proposed standardized object-oriented expression language for clinical decision support. Medinfo 11 (Pt 1), 164–168.

Sullivan, S.D., Carlson, J.J., Hansen, R.N., 2013. Comparative effectiveness research in the United States: a progress report. J. Med. Econ. 16 (2), 295–297.

Tu, S.W., Musen, M.A., Shankar, R., Campbell, J., Hrabak, K., McClay, J., et al., 2004. Modeling guidelines for integration into clinical workflow. Medinfo 11 (Pt 1), 174–178.

USPSTF, 2013. U.S. Preventive Services Task Force website. Retrieved October 21, 2013, from: <http://www.uspreventiveservicestaskforce.org/>.

Wang, D., Peleg, M., Bu, D., Cantor, M., Landesberg, G., Lunenfeld, E., et al., 2003. GESDOR – a generic execution model for sharing of computer-interpretable clinical practice guidelines. AMIA. Annu. Symp. Proc. 694–698.

Warner, H.R., Toronto, A.F., Veasy, L.G., 1964. Experience with Bayes' Theorem for Computer Diagnosis of Congenital Heart Disease. Ann. N. Y. Acad. Sci. 115, 558–567.

Weaver, R.R., 2003. Informatics tools and medical communication: patient perspectives of "knowledge coupling" in primary care. Health. Commun. 15 (1), 59–78.

Yu, H., Cao, Y.G., 2008. Automatically extracting information needs from ad hoc clinical questions. AMIA. Annu. Symp. Proc. 96–100.