

Software Engineering for Health Care and Biomedicine

Adam B. Wilcox, David K. Vawdrey, and Kensaku Kawamoto

Contents

- 6.1 How Can a Computer System Help in Health Care? – 178**
- 6.2 Software Functions in Health Care – 178**
 - 6.2.1 Case Study of Health Care Software – 178
 - 6.2.2 Acquiring and Storing Data – 180
 - 6.2.3 Summarizing and Displaying Data – 181
 - 6.2.4 Facilitating Communication and Information Exchange – 182
 - 6.2.5 Generating Alerts, Reminders, and Other Forms of Decision Support – 182
 - 6.2.6 Supporting Educational, Research, and Population and Public Health Initiatives – 183
- 6.3 Software Development and Engineering – 183**
 - 6.3.1 Software Development – 183
 - 6.3.2 Software Development Models – 188
 - 6.3.3 Software Engineering – 190
- 6.4 Emerging Influences and Issues – 199**
- 6.5 Summary – 201**
- References – 203**

Learning Objectives

After reading this chapter, you should know the answers to these questions:

- What key functions do software applications perform in health care?
- How are the components of the software development lifecycle applied to health care?
- What are the trade-offs between purchasing commercial, off-the-shelf systems and developing custom applications?
- What are important considerations in comparing commercial software products?
- Why do systems in health care, both internally-developed and commercial purchased, require continued software development?

6.1 How Can a Computer System Help in Health Care?

In this chapter, we focus on the software applications and components of health care information systems, and describe how they are used and applied to support health care delivery. We give examples of some basic functions that may be performed by health information systems, and discuss important considerations in how the software may be acquired, implemented and used. This understanding of how a system gets put to use in health care settings will help as you read about the various specific applications in the chapters that follow.

Health care is an information-intensive field. Clinicians are constantly gathering, reviewing, analyzing and communicating information from many sources to make decisions. Humans are complex, and individuals have many different characteristics that are relevant to health care and that need to be considered in decision-making. Health care is also complex, with a huge body of existing knowledge that is expanding at an ever-increasing rate. Software for managing health information is intended to facilitate the use of this information at various points in the

care delivery process. Software can determine the ways by which data are obtained, organized and processed to yield information. Software, in terms of design, development, acquisition, configuration and maintenance, is therefore a major component of the field of biomedical informatics. This chapter provides an introduction to some of the practical considerations regarding health information software, including both general software engineering principles, as well as the application of these principles to health care settings.

To this aim, we first describe the major software functions within a health care environment or health information system. While not all functions can be covered in detail, some specific examples are given to indicate the breadth of software applications as well as to provide an understanding of their relevance. We also describe the software development life cycle, with specific applications to health care. We then describe important considerations and strategies for acquiring and implementing software in health care settings. Finally, we discuss emerging trends influencing software engineering related to health information systems. Each system can be considered in regard to what it would take to make it functional in a health care system, and what advantages and disadvantages the software may have, based on how it was created and implemented. Understanding these principles will help you identify the risks and benefits of various applications, so that you can identify how to optimize the positive impact of health information systems.

6.2 Software Functions in Health Care

6.2.1 Case Study of Health Care Software

The following case study illustrates many important functions of health care software.

John Miller is a 42-year old man living in a medium-sized U.S. city. He is married and has

two children. He has type 2 diabetes, but it is currently well controlled and he has no other health concerns. There is some history of cardiovascular disease in his family. John has a primary care physician, Linda Stark, who practices at a clinic that is part of a larger health delivery network, Generation Healthcare System (GHS). GHS includes a physician group, primary and specialty care clinics, a tertiary care hospital and an affiliated health insurance plan.

John needs to make an appointment with Dr. Stark. He logs into the GHS **patient portal** and uses an online scheduling application to request an appointment. While in the patient portal, John also reviews results from his most recent visit and prints a copy of his current medication list in order to discuss the addition of an over-the-counter supplement he recently started taking.

Before John arrives for his visit, the clinic's scheduling system has already alerted the staff of John's appointment and the need to collect information related to his diabetes. Upon his arrival, Dr. Stark's nurse gathers the requested diabetes information and other vital signs data and enters these into the **electronic health record** (EHR). In the exam room, Dr. Stark reviews John's history, the new information gathered during this visit, and recommendations and reminders provided by the EHR on a report tailored to her patient's medical history. They both go over John's medication list and Dr. Stark notes that, according to the EHR's drug-drug interaction tool, the supplement he is taking may have an interaction with one of his diabetes medications. One of the reminders suggests that John is due for a hemoglobin A1c (HbA1c) test, and Dr. Stark orders this in the EHR. Dr. Stark's nurse, who has been notified of the lab test order, draws a blood sample from John. Before the appointment ends, Dr. Stark completes and signs the clinic note and forwards a visit summary for John to review on the patient portal.

A few days after his appointment, John receives an email from GHS that notifies him of an important piece of new information in his patient record. Logging into the patient portal application, John sees that his HbA1c test is back. The test indicates that the result is elevated. Dr. Stark has added a note to the result

saying that she has reviewed the lab and would like to refer John to the GHS Diabetes Specialty Clinic for additional follow-up. John uses the messaging feature in the patient portal to respond to Dr. Stark and arrange for an appointment. John also clicks on an **info button** next to the lab result to obtain more information about the abnormal value. He is linked to patient-focused material about HbA1c testing, common causes for elevated results, and ways this might be addressed. Lastly, John reviews the visit summary note from his appointment with Dr. Stark to remind him about suggestions she had for replacing his supplement.

At his appointment with the Diabetes Specialty Clinic, John notes that they have access to all the information in his record. A diabetes care manager, Maria, reviews the important aspects of John's medical history. She suggests more frequent monitoring of his laboratory test results and evaluating whether he is able to control his diabetes without changes to his medications. Maria highlights diet and exercise suggestions in his patient portal record that have been shown to help similar patients. When the visit is complete, Maria sends an electronic summary of the visit to Dr. Stark.

A year later, John is experiencing greater difficulty controlling his diabetes. Dr. Stark and Maria have continued to actively monitor his HbA1c and other laboratory test results, and occasionally make changes to his treatment regimen. They use the EHR to visualize laboratory test results and correlate them with changes in medications. Due to a variety of personal and financial challenges, John struggles with adherence to his medication regimen, and he is not maintaining a healthy diet. As a result, his blood sugar has become seriously unstable, and the population health management module of the EHR flags John for urgent evaluation due to a dangerously high home blood glucose reading. Maria confirms the reading with John, collects additional information about his health status, and escalates the issue to Dr. Stark. Dr. Stark then recommends John go to the GHS hospital emergency department (ED) for urgent evaluation. Doctors in the ED access John's electronic record including his medication and lab history, as well as notes from Dr. Stark and Maria, which help them quickly assess his condition

and develop a treatment plan. John is admitted to the hospital, and physicians, nurses, and others caring for him access his longitudinal medical records and document new observations and treatments. They are also able to electronically reconcile his outpatient prescriptions with his inpatient medications to ensure continuity. After a brief hospital stay, John is stabilized and ready to be discharged, with an updated list of medications.

Because Dr. Stark is listed as John's primary care physician, she is notified electronically of both the hospital admission and discharge. She reviews his discharge summary in the EHR and instructs her staff to send a message through the patient portal to John, to let him know she reviewed his inpatient record and to schedule a follow-up appointment.

The GMS EHR is also part of a statewide **health information exchange (HIE)**, which allows medical records to be easily shared with health care providers outside the GMS system. This means that if John should need to visit a hospital, emergency department or specialty care clinic outside the GMS network, his record would be available for review and any information entered by these outside providers would be similarly available to Dr. Stark and others within the GMS network. The local and state health departments where John lives are also linked to the HIE. This allows clinics, hospitals and labs to electronically submit information to the health departments for disease surveillance and case reporting purposes.

Back at home, John's wife, Gina, is able to view his medical records on the GHS patient portal because he has granted her proxy access to his account. This allows her to see notes from Dr. Stark and schedule appointments. Gina also views the hospital discharge instructions that were electronically sent to John's patient record. As she reviews the information about diabetes that GHS had automatically linked to John's record, Gina sees a notification about a clinical research study involving genetic links with diabetes. Concerned about their two children, Gina discusses the study with John, and they review more online materials about the study. Interested in the possible benefits of the research, John electronically volunteers to participate in the study, and he is later contacted

by a study coordinator. Because GHS investigators are conducting the study, relevant parts of John's EHR are easily shared with the clinical trials management system.

This fictional case study highlights many of the current opportunities for improving health care delivery, including improved access to care, increased patient engagement, shared patient-provider decision-making, better care management, medication reconciliation, improved transitions of care, population health management, and research recruitment. In the case study, each of these goals required software to make health information accessible to the correct individuals at the proper time.

In today's health care system, few individuals enjoy the interaction with software depicted in the *John Miller* case study. Although the functions described in the scenario exist at varying levels of maturity, most health care delivery institutions have not connected all the functions together as described. The current role of software engineering in health care is therefore two-fold: to design and implement software applications that provide required functions, and to connect these functions in a seamless experience for both the clinicians and the patients.

The case study highlights the usefulness of several functions provided by health care software applications for clinicians, patients, and administrators. Some of these functions include:

1. Acquiring and storing data
2. Summarizing and displaying data
3. Facilitating communication and information exchange
4. Generating alerts, reminders, and other forms of decision support
5. Supporting educational, research, and public health initiatives

6.2.2 Acquiring and Storing Data

The amount of data needed to describe the health and health care of even a single person is huge. Health professionals require assistance with data acquisition to deal with the data that must be collected and processed. One of the first uses of computers in a medi-

cal setting was the automatic analysis of blood specimens and other body fluids by instruments that measure chemical concentrations or that count cells and organisms. These systems generated printed or electronic results to health care workers and identified values that were outside normal limits. Computer-based patient monitoring that collected physiological data directly from patients were another early application of computing technology (see ► Chap. 19). These systems provided frequent, consistent collection of vital signs, electrocardiograms (ECGs), and other indicators of patient status. More recently, researchers have developed medical imaging applications as described in ► Chaps. 9 and 20, including computed tomography (CT), magnetic resonance imaging (MRI), and digital subtraction angiography. The calculations for these computationally intensive applications cannot be performed manually; computers are required to collect and manipulate millions of individual observations.

Early computer-based medical instruments and measurement devices provided results only to human beings. Today, most instruments can transmit data directly into the EHR, although the interfaces can still be awkward and poorly standardized (see ► Chaps. 5 and 8). Computer-based systems that acquire information directly from patients are also data-acquisition systems; they free health professionals from the need to collect and enter demographic and health history information.

Various departments within a hospital use computer systems to store clinical data. For instance, clinical laboratories use information systems to keep track of orders and specimens and to report test results; most pharmacy and radiology departments use computers to perform analogous functions. Their systems may connect to outside services (e.g., pharmacy systems are typically connected to one or more drug distributors so that ordering and delivery are rapid and local inventories can be kept small). By automating processing in areas such as these, health care facilities are able to provide efficient service, reduce labor costs, and minimize errors.

6.2.3 Summarizing and Displaying Data

Computers are well suited to performing tedious and repetitive data-processing tasks, such as collecting and tabulating data, combining related data, and formatting and producing reports. They are particularly useful for processing large volumes of data.

Data acquired by computer systems can be detailed and voluminous. Data analysis systems must aid decision makers by reducing and presenting the intrinsic information in a clear and understandable form. Software can be used to create useful visualizations that facilitate trend analysis and compute secondary parameters (e.g., means, standard deviations, rates of change) to help identify abnormalities. Clinical research systems have modules for performing powerful statistical analyses over large sets of patient data. When employing such tools, research investigators should have insight into the methods being used. For clinicians, graphical displays are useful for interpreting data and identifying trends.

Fast retrieval of information is essential for any computer system. Data must be well organized and indexed so that information recorded in an EHR system can be easily retrieved. Here the variety of users must be considered. Obtaining recent information about a patient entering the office differs from the needs that a research investigator will have in accessing the same data. The query interfaces provided by EHRs and clinical research systems assist researchers in retrieving pertinent records from the huge volume of patient information. Recently, there has been increasing industry adoption of the **Health Level 7 International (HL7)** Fast Healthcare Interoperability Resources (FHIR) standard for sharing data on a patient-by-patient basis. The FHIR standard is being adapted for population-level data sharing through the FHIR Bulk Data initiative. As discussed in ► Chap. 21, bibliographic retrieval systems are also an essential component of health information services.

6.2.4 Facilitating Communication and Information Exchange

In hospitals and other large-scale health care institutions, myriad data are collected by multiple health professionals who work in a variety of settings; each patient receives care from a host of providers—nurses, physicians, technicians, pharmacists, and so on. Communication among the members of the team is essential for effective health care delivery. Data must be available to decision makers when and where they are needed, independent of when and where they were obtained. Computers help by storing, transmitting, sharing, and displaying those data. As described in ► Chaps. 2 and 12, the patient record is the primary vehicle for communication of clinical information. The limitation of the traditional paper-based patient record is the concentration of information in a single location, which prohibits simultaneous entry and access by multiple people. Hospital information systems (HISs; see ► Chap. 13) and EHR systems (► Chap. 12) allow distribution of many activities, such as admission, appointment, and resource scheduling; review of laboratory test results; and inspection of patient records to the appropriate sites.

Information necessary for specific decision-making tasks is rarely available within a single computer system. Clinical systems are installed and updated when needed, available, and affordable. Furthermore, in many institutions, inpatient, outpatient, and financial activities are supported by separate organizational units. Patient treatment decisions require inpatient and outpatient information. Hospital administrators must integrate clinical and financial information to analyze costs and to evaluate the efficiency of health care delivery. Similarly, clinicians may need to review data collected at other health care institutions, or they may wish to consult published biomedical information. Communication networks that permit sharing of information among independent computers and geographically distributed sites are now widely available. Actual integration of the information they contain requires additional software,

adherence to standards, and operational staff to keep it all working as technology and systems evolve.

6.2.5 Generating Alerts, Reminders, and Other Forms of Decision Support

In the end, all the functions of storing, displaying and transmitting data support decision-making by health professionals, patients, and their caregivers. The distinction between decision-support systems and systems that monitor events and issue alerts is not clear-cut; the two differ primarily in the degree to which they interpret data and recommend patient-specific action. Perhaps the best-known examples of decision-support systems are the clinical consultation systems or event-monitoring systems that use population statistics or encode expert knowledge to assist physicians in diagnosis and treatment planning (see ► Chap. 22). Similarly, some nursing information systems help nurses to evaluate the needs of individual patients and thus assist their users in allocating nursing resources. ► Chapter 22 discusses systems that use algorithmic, statistical, or artificial-intelligence (AI) techniques to provide advice about patient care.

Timely reactions to data are crucial for quality in health care, especially when a patient has unexpected problems. Data overload, created by the ubiquity of information technology, is as detrimental to good decision making as is data insufficiency. Data indicating a need for action may be available but are easily overlooked by overloaded health professionals. Surveillance and monitoring systems can help people cope with all the data relevant to patient management by calling attention to significant events or situations, for example, by reminding doctors of the need to order screening tests and other preventive measures (see ► Chaps. 12 and 22) or by warning them when a dangerous event or constellation of events has occurred.

Laboratory systems routinely identify and flag abnormal test results. Similarly, when

patient-monitoring systems in intensive care units detect abnormalities in patient status, they sound alarms to alert nurses and physicians to potentially dangerous changes. A pharmacy system that maintains computer-based drug-profile records for patients can screen incoming drug orders and warn physicians who order a drug that interacts with another drug that the patient is receiving or a drug to which the patient has a known allergy or sensitivity. By correlating data from multiple sources, an integrated clinical information system can monitor for complex events, such as interactions among patient diagnosis, drug regimen, and physiological status (indicated by laboratory test results). For instance, a change in cholesterol level can be due to prednisone given to an arthritic patient and may not indicate a dietary problem.

6.2.6 Supporting Educational, Research, and Population and Public Health Initiatives

Rapid growth in biomedical knowledge and in the complexity of therapy management has produced an environment in which students cannot learn all they need to know during training—they must learn how to learn and must make a lifelong educational commitment. Today, physicians and nurses have available a broad selection of computer programs designed to help them to acquire and maintain the knowledge and skills they need to care for their patients. The simplest programs are of the drill-and-practice variety; more sophisticated programs can help students to learn complex problem-solving skills, such as diagnosis and therapy management (see ► Chap. 21). Computer-aided instruction provides a valuable means by which health professionals can gain experience and learn from mistakes without endangering actual patients. Clinical decision-support systems and other systems that can explain their recommendations also perform an educational function. In the context of real patient cases, they can suggest actions and explain the reasons for those actions.

As health care increasingly shifts to a mode of care based on population health rather than episodic health care transactions, there is increasing need for information systems to monitor and manage individuals' health outside the context of clinical visits. Surveillance also extends beyond the health care setting. Appearances of new infectious diseases, unexpected reactions to new medications, and environmental effects should be monitored. Thus the issue of data integration has a national or global scope (see the discussion of the National Health Information Infrastructure in ► Chaps. 1 and 16 that deals with public health informatics).

6.3 Software Development and Engineering

Clearly, software can be used in many different ways to manage and manipulate health information to facilitate health care delivery. However, just using a computer or a software program does not improve care. If critical information is unavailable, or if processes are not organized to operate smoothly, a computer program will only expose challenges and waste time of clinical staff that could be better applied in delivering care. To be useful, software must be developed with an understanding of its role in the care setting, geared to the specific functions that are required, and developed correctly. To be used, software must be integrated to support the users' workflow. We will discuss both aspects of software engineering – development and integration.

6.3.1 Software Development

Software development can be a complex, resource-intensive undertaking, particularly in environments like health care where safety and security provide added risk. The **software development life cycle** (SDLC) is a framework imposed over software development in order to better ensure a repeatable, predictable process that controls cost and improves quality of

the software product (usually an application). SDLC is a subset of the systems development life cycle, focusing on the software component of a larger system. In practice, and particularly in health care, software development encompasses more than just the software, often stretching into areas such as process re-engineering in order to maximize the benefits of the software product. Although SDLC most literally applies to an in-house development project, all or most of the life cycle framework is also relevant to shared development and even purchase of commercial off-the-shelf (COTS) software. The following is an overview of the phases of the SDLC.

6.3.1.1 Planning/Analysis

The software development life cycle begins with the formation of a project goal during the planning phase. This goal typically derives from an organization's or department's mission/vision, focusing on a particularly need or outcome. This is sometimes called project conceptualization. Planning includes some initial scoping of the project as well as resource identification (including funding). It is important to address what is not included in the project in order to create appropriate expectations for the final product. A detailed analysis of current processes and needs of the target users is often done. As part of the analysis, specific user requirements are gathered. Depending on the development process, this might include either detailed instructions on specific functions and operating parameters or more general user stories that explain in simple narrative the needs, expected workflow and outcomes for the software. It is important that users of the system are consulted, as well as those in the organization who will implement and maintain the software. The decision of whether to develop the software in-house, partner with a developer, or purchase a vendor system will likely determine the level of detail needed in the requirements. Vendors will want very specific requirements that allow them to properly scope and price their work. The requirements document will usually become part of a contract with a vendor and will be used to determine if the final product meets the agreed specification for the soft-

ware. In-house development can have less detailed requirements, as the contract to build the software is with the organization itself, and can allow some evolution of the requirements as the project progresses. However, the more flexibility that is allowed and the longer changes or enhancements are permitted, the higher the likelihood of "scope creep" causing schedule and cost overruns.

Other tasks performed during analysis include an examination of existing products and potential alternative solutions, and, particularly for large projects, a cost/benefit analysis. A significant, and frequently overlooked, aspect of the planning and analysis phase is to determine outcome measures that can be used during the life cycle to demonstrate progress and evaluate success or failure of the project. These measures can be refined and details added as the project progresses. The planning and analysis phase typically ends when a decision to proceed is made, along with at least a rough plan of how to implement the next steps in the SDLC. If the organization decides to purchase a solution, a request for proposals (RFP) that contains the requirements document is released to the vendor community.

The planning and analysis stage of software development is perhaps both the most difficult and the most important stage in the development lifecycle as it is applied to health care. Requirements for software in health care are inherently difficult to define for many reasons. Health care practice is constantly changing, and as new therapies or approaches are discovered and validated, these new advancements can change the way care is delivered. In addition, the end-users of health care software are comparatively advanced relative to other industries. Unlike industries where front-line workers may be directed by supervisors with more advanced training and greater flexibility in decision making, in health care the front-line workers are often physicians, who are often the most highly-trained workers in the system (although not necessarily the most advanced with respect to computer literacy) and require the greatest flexibility for decisions. This flexibility makes it difficult to define workflows or even get indications of the workflows being followed,

since physicians will not always make explicit what actions or plans are being pursued. This flexibility is important for patient care, because it allows front-line clinicians to adapt appropriately to different settings, staffing levels, and specialties. The need for flexibility is such that defining requirements for software that could reduce flexibility is criticized as “cookbook” medicine, constituting a common reason for resistance to software adoption. However, this resistance is not just characteristic of software – clinical guidelines and other approaches to structured or formalized care processes can also be criticized, and the challenge of applying discovered-knowledge to clinical care processes remains difficult.

Over time, however, there have been some successful efforts that have defined standard requirements for health information software. Among the most notable efforts have been in EHRs, where organizations have created lists of requirements and certified systems that match those requirements. The Certification Commission for Health Information Technology (CCHIT) began in 2004 and defined criteria for electronic health records’ functionality, interoperability and security (Leavitt and Gallagher 2006). Later, the certification approach was adopted by the Office of the National Coordinator of Health Information Technology (ONC) in 2010, when a list of EHR functions that were most related to “meaningful use” of EHRs was established (Blumenthal and Tavenner 2010). Such “meaningful use” of EHRs came with significant financial incentives administered by the Centers for Medicare and Medicaid Services (CMS), leading to a rapid increase in the adoption of EHRs meeting these requirements (Washington et al. 2017).

6.3.1.2 Design

During the Design phase, potential software solutions are explored. System architectures are examined for their abilities to meet the needs stated in the requirements. Data storage and interface technologies are assessed for appropriate fit. User front-end solutions are investigated to assess capabilities for required user input and data display functions. Other

details, such as security, performance, and internationalization are also addressed during design. Analysts with domain knowledge in the target environment are often employed during this phase in order to translate user requirements into suitable proposals. Simple mock-ups of the proposed system may be developed, particularly for user-facing components, in order to validate the design and identify potential problems and missing information. Closely related to this, an integrated, automated testing architecture, with appropriate testing scripts/procedures, may be designed in this phase in order to ensure the software being developed meets quality standards and is responsive to the requirements. The depth and completeness of the design is contingent on the software development process, as well as other factors. In some cases, the entire design is completed before moving on to software coding. In other development strategies, a high-level system architecture is designed but the details of the software components are delayed until each component or component feature is being created. The pros and cons of these approaches are discussed later in this chapter. For vendor-developed systems, the purchasing organization will often hold design reviews and demonstrations of mock-ups or prototypes with the vendor to assess the solutions. In the case of COTS software, the purchasing organization relies on the vendor’s system description and reviews from third parties, supplemented by system demonstrations, to determine the appropriateness of the design. As with the Analysis phase, it is important to include the target users and IT operations personnel in the design reviews.

Ideally, the software could be designed solely around the care requirements and the use of information. However, rarely are the clinical requirements of the use case the only consideration. In the design phase, other requirements are considered, such as the software cost and how it integrates with an existing health IT strategy of an organization. Resources applied to a development project are not available for other potential projects, so costs are always influential. The design phase must consider various alternatives to

meet the most important requirements, recognizing trade-offs and contingency approaches. Additional considerations are how the software will support long-term needs, not just the immediate requirements that have been identified. Clinicians and clinical workflow analysts are often the primary participants in the requirements analysis stage, whereas informaticians are more prominent in the design phase. This is because during this latter phase the clinical goals and strategies are considered together with what can be vastly different design approaches, and the ability to consider the various strengths and weaknesses of these different approaches is critical. Often, design considerations are between custom development, purchasing niche applications, or purchasing components of a monolithic EHR. The considerations of development versus COTS software is discussed in more detail in the Acquisition Strategy section (► 6.3.3.1) below.

6.3.1.3 Development

Coding of the software is done during the Development phase of the SDLC. The software engineers use the requirements and system designs as they program the code. Analysts help resolve questions about requirements and designs for the programmers when it is unclear how software might address a particular feature. The software process defines the pace and granularity of the development. In some cases, an entire software component or system is developed at once by the team. In other cases, the software is broken down into logical pieces and the programmers only work on the features that are relevant to the piece they are currently working on. As software components are completed, unit tests are run to confirm the component is free of known bugs and produces expected outputs or results.

In health care, development includes coding of custom software as well as configuration of COTS software. Health care practices across institutions (and even within larger organizations) are so variable that all software requires some – often substantial – configuration. Configuration can range from assigning local values to generic variables within the

software, to complete development of documentation templates, order sets, clinical decision support rule, reports, and so on. In fact, configuration can be so considerable that institutions may use an internal brand name for the software and configuration project that is different from the name of the COTS software, which represents their local configuration. This configuration is often done using tools built specifically for the commercial software, which facilitate the integration of the configuration products into the software infrastructure. The tools can be complex, requiring significant training for developers. Typically, tools work well for basic configuration and may also have advanced functionality that can be used to configure more complicated functionality. The most intensive time investment for configuration is typically when the tools do not directly support certain configurations, and developers must find approaches to creatively adapt the development “around the tools.”

6.3.1.4 Integration and Test

For complex software projects consisting of several components and/or interfaces with outside systems, an Integration phase in the SDLC is employed to tie together the various pieces. Some aspects of the software integration are likely done during the Development phase by simulating or mocking the outputs to, and inputs from, other systems. During Integration, these connections are finalized. Simulations are run to demonstrate functional integration of the various system components. Once the various components are integrated, a thorough testing regimen is conducted in order to prove the end-to-end operation of the entire software system. Specific test scenarios are run with known inputs and expected outputs. This is typically done in a safe, non-operational environment in order to avoid conflicts and unnecessary risk to production environments, although some inbound information from live systems may be used to verify scenarios that are difficult to simulate.

Testing and integration in health care are similar to other complex environments, in that it can be difficult to create a testing environment that matches the dynamics of the

real-world setting. Generally, testing is done around multiple use cases or case studies, using data to support the cases. In a production environment, however, there may be data and information that do not match the case studies, since both people and health care are complex. As a result, internally-developed applications are often provisionally used in a “pilot” phase as part of testing. For COTS software, companies may use simulation laboratories that try to mimic the clinical environment, or work with specific health care organizations as development and testing partners. Later, however, this can lead to challenges if data representing the dynamics of one organization are not easily transferable, and software must be further tested with new environments. Software transferability between institutions has been demonstrated in studies, even for specific applications (Hripcsak et al. 1998). Another challenge is that with current privacy laws, organizations are more reluctant to release data to vendors for testing.

6.3.1.5 Implementation

Once the software passes integration testing it moves to the implementation phase. In this phase, the software is installed in the live environment. In preparation for installation, server hardware, user devices, network infrastructure, changes specific to individual facilities, etc., may need to be implemented and tested as well. In addition, user training will be performed in the weeks before the software goes live. Any changes to policies and procedures required by the software will also be implemented in the build-up to installation.

Health care presents interesting considerations in each phase of the software development cycle, but the challenges have been more visible in implementation than any other phase. This may be because health IT, while intended to facilitate more efficient workflows with information, is still disruptive. Disruption happens most during implementation, when clinicians actually begin using the software, and studies have shown that during this time clinical productivity does decline (Shekelle et al. 2006). If users do not perceive that the benefits are sufficient to justify this disrupt-

tion, or if the efficiency does not improve quickly enough after the initial implementation, they may choose to disregard the software or even revolt against its implementation. There have been prominent examples in biomedical informatics of software implementations failing during implementation (Bates 2006; Smelcer et al. 2009; Sullivan 2017), and even studies demonstrating harm (Han et al. 2005). Because of these risks, health IT professionals need to be flexible in implementation, and adapt the implementation strategies to how the system is adopted. Users have been shown to use health IT software in different ways for different benefits, and may need incentives or prodding to advance to different levels of use.

6.3.1.6 Verification and Validation

To ensure that the software satisfies the original requirements for the system and meets the need of the organization, a formal verification and validation of the software is performed. The implementing organization *verifies* that the software has the features and performs all the functions specified in the requirements document. The software is also *validated* to show that it performs according to specified operational requirements, that it produces valid outputs, and that it can be operated in a safe manner. For purchased software, the verification and validation phase is used by the purchasing organization in order to officially accept the software.

Since clinicians often use software at different levels or in different ways, tracking patterns of use can be an important approach for verification and validation of software in health care. Additionally, because they have experience working in complicated environments, users can be good at identifying inconsistencies in data or software functions. Two approaches that have been used and can be successful for validation are monitoring use, and facilitating user feedback.

6.3.1.7 Operations and Maintenance

Software eventually enters an operations and maintenance (O&M) phase where it is being regularly used to support the operational needs of the organization. During this phase,

an O&M team will ensure that the software is operating as desired and will be fielding the support needs of the users. Updates may need to be installed as new versions of the software are released. This may require new integration and testing, implementation, and verification and validation steps. Ongoing training will be required for new users and system updates. The O&M team may conduct regular security reviews of the system and its use. Data repositories and software interfaces will be monitored for proper operation and continued information validity. Software bugs and feature enhancement requests will be collected. These may drive an entire new development life cycle as new requirements persuade an organization to explore significant upgrades to its current software or even an entirely new system.

Maintenance is a demanding task in health information software. It involves correcting errors; adapting configurations and software to growth, new standards, and new regulations; and linking to other information sources. Maintenance tasks can exceed by more than double the initial acquisition costs, making it a substantial consideration that should affect software design. COTS suppliers often provide maintenance services for 15–30% of the purchase price annually, but custom development or configuration maintenance must be supported by the purchasing organization. If the software is not maintained, it can quickly become unusable in a health care setting. Indeed, optimization of COTS EHRs is a central and ongoing focus of applied clinical informatics, and this is likely to continue for the foreseeable future.

6.3.1.8 Evaluation

An important enhancement to the SDLC suggested by Thompson et al. (1999) is the inclusion of an evaluation process during each of the phases of the life cycle. The evaluation is influenced by risk factors that may affect a particular SDLC segment. An organization might perform formative evaluations during each phase, depending on specific needs, in order to assess the inputs, processes and resources employed during development. During


Verification and Validation or O&M, a summative evaluation may be performed to assess the outcome effects, organizational impact, and cost-benefit of the software solution.

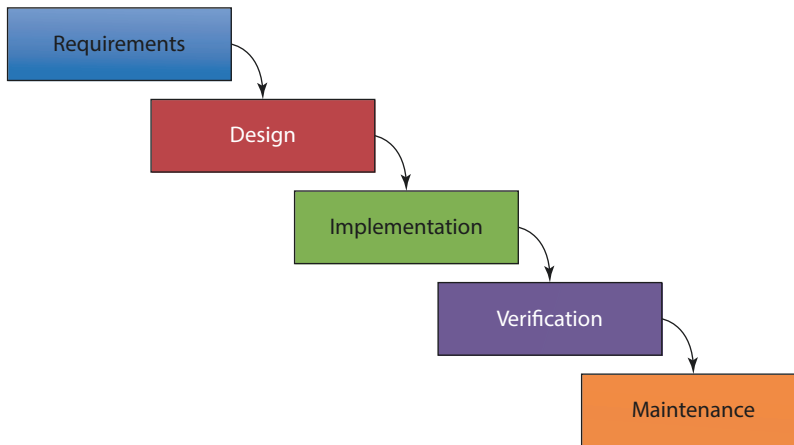
Health IT is considered an intervention into the health care delivery system, so evaluations have been done and published as comparative studies in the clinical literature (Bates et al. 1998; Campanella et al. 2016; Evans et al. 1998; Hunt et al. 1998; Jones et al. 2014). These evaluations, and syntheses of multiple studies, have identified areas of impact and areas where the effect of health IT software is inconsistent. Researchers have also noted that most of these studies have occurred in institutions where software was developed internally, with disproportionate under-representation of COTS software systems in evaluations, especially considering that most health care institutions use COTS rather than internal development (Chaudhry et al. 2006). It is hoped that the existing evaluations can be a model for software evaluations of COTS, to clarify their impact on care.

6.3.2 Software Development Models

Different software development processes or methods can be used in an SDLC. The **software development process** describes the day-to-day methodology followed by the development team, while the life cycle describes a higher-level view that encompasses aspects that take place well before code is ever written and after an application is in use. The following are two of the most common examples of different development processes in clinical information systems development.

6.3.2.1 Waterfall Model

The Waterfall model of software development suggests that each step in the process happens sequentially, as shown in  Fig. 6.1. The term “Waterfall” refers to the analogy of water cascading downward in stages. A central concept of the Waterfall methodology is to solidify all of the requirements, establish complete functional specifications, and create the final soft-



■ Fig. 6.1 The Waterfall model of software development

ware design prior to performing programming tasks. This concept is referred to as “Big Design Up Front,” and reflects the thinking that time spent early-on making sure requirements and design are correct saves considerable time and effort later. Steve McConnell, an expert in software development, estimated that “...a requirements defect that is left undetected until construction or maintenance will cost 50–200 times as much to fix as it would have cost to fix at requirements time” (McConnell 1996).

The waterfall model provides a structured, linear approach that is easy to understand. Application of the model is best suited to software projects with stable requirements that can be completely designed in advance. In practice, it may not be possible to create a complete design for software a priori. Requirements and design specifications can change even late in the development process. Clients may not know exactly what requirements they need before reviewing a working prototype. In other cases, software developers may identify problems during the implementation that necessitate reworking the design or modifying the requirements.

6.3.2.2 Agile Models

In contrast to the Waterfall model, modern software development approaches have attempted to provide more flexibility, particularly in terms of involving the customer

throughout the process. In 2001, a group of software developers published the Manifesto for Agile Software Development, which emphasizes iterative, incremental development and welcomes changes to software requirements even late in the development process (Beck et al. 2001).

Agile development eschews long-term planning in favor of short iterations that usually last from 1 to 4 weeks. During each iteration, a small collaborative team (typically 5–10 people) conducts planning, requirements analysis, design, coding, unit testing, and acceptance testing activities with direct involvement of a customer representative. Multiple iterations are required to release a product, and larger development efforts involve several small teams working toward a common goal. The agile method is value-driven, meaning that customers set priorities at the beginning of each iteration based on perceived business value.

Agile methods emphasize face-to-face communication over written documents. Frequent communication exposes problems as they arise during the development process. Typically, a formal meeting is held each morning during which team members report to each other what they did the previous day, what they intend to do today, and what their roadblocks are. The brief meeting, sometimes called a “stand-up,” “scrum,” or “huddle,” usually lasts 5–15 minutes, and includes the

development team, customer representatives and other stakeholders. A common implementation of agile development is Extreme Programming.

6.3.3 Software Engineering

The software development life cycle can be used to actually create the software, and understanding it is critical for those developing software in biomedical informatics. However, as the field has expanded, software has matured to the point that it is developed by and available from commercial companies, so that software development has become less of a concern for most of the field. A more important consideration in biomedical informatics has been the strategy of whether to develop and how to develop. Software vendors can spread development costs over multiple organizations, rather than one organization having to fund the full development, which can make purchasing software economically advantageous. On the other hand, the core requirements for software continue to change, and sometimes organizations need specific capabilities that are not met by existing vendor software options. In addition to software development, informaticians often participate in software acquisition, as well as in subsequent enhancements to acquired software.

6.3.3.1 Software Acquisition

In health care information technology applications, a significant question is whether to develop the software internally or purchase an existing system from a vendor. Whether to “build vs. buy” is a core decision in planning and implementation.

Considerations for purchasing software begin with how the software will be selected. Software can be a component of a monolithic vendor system, be a secondary application sold by the same vendor as the EHR, or be “best-of-breed,” meaning the software that meets the requirements best, independent of its architecture or source. Another consideration is whether the software needs to integrate with other applications. Some specialty

applications require minimal data sharing with other software, while other applications must be tightly integrated with existing systems to achieve a benefit. Two examples are a picture archiving and communications system (PACS) and a medication reconciliation tool. Perhaps the most important requirement for a PACS is to provide access to images for a radiologist, who can then “read” the image and document a report which can be transferred into the EHR as a static document. On the other hand, a medication reconciliation tool may need substantial integration with medication ordering and administration modules in an EHR to support workflows of the care team. Another consideration, related to integration, is the storage mechanism. A stand-alone system will likely have a separate database, while an integrated system may be able to store and retrieve data using a common data repository. User interface deployment is also important, and possibilities include Web-based clients, **thin clients** (e.g., Citrix), and locally-installed **thick-client** applications. Greater functionality may exist with a thick-client application, but Web-based and thin-client tools are easier to update and distribute to users. Finally, security and privacy considerations are critical in health care, and can influence both the requirements and design of software. Security considerations can include whether **user authentication** is shared with other applications, or what data access events are audited for identifying potential security threats.

Most healthcare delivery organizations today use commercial – as opposed to locally developed – EHRs. But in reality, there is still a mix between building and buying health information technology. As mentioned, organizations using commercial systems require substantial local configuration that ranges from application-specific parameter configuration to coordinating multiple software applications to link together. Even when there is a commitment to limit local configuration, there may still be separate systems, local configuration or even development with data warehousing and analytics solutions for the EHR data. There is no single solution, commercial or internally-developed, that meets all

the health information needs of most health care organizations, and many implementations involve a mixture of software from multiple vendors. While there can be advantages to allowing best-of-breed, a current trend among organizations is to consolidate as much functionality as possible with one vendor. Another observed trend is for organizations that build systems to consider purchasing COTS, due to the substantial maintenance costs associated with in-house development and the increased functionality often available with vendor solutions. At present, virtually all health care organizations that utilize an EHR in the United States use a COTS solution or are in the process of migrating to such a solution.

Usually, if vendor software exists, it is more cost-efficient to purchase the software than build comparable capabilities internally for use at a single organization. This is because the vendor can spread development costs over multiple organizations, rather than one organization having to fund the full development. In fact, few organizations have the existing infrastructure and personnel to consider internal development for anything other than small applications. However, those few institutions with developed health information systems are notable for the success of their software. So while the costs may be higher for internal development, the benefits may also be higher. Furthermore, such solutions may be potentially licensed to other organizations, thereby spreading the cost of development across multiple organizations. Still, these institutions have invested many years to build an infrastructure that makes these benefits possible, and it is unlikely that many organizations can afford the time and resource investment to follow the same model. Even within historically internally-developing organizations, buying systems that can integrate with the existing system is oftentimes more efficient than development. An appropriate general guide is therefore, “Buy where you can, build where you can’t.”

Once an organization decides to acquire a health information system, there are many other decisions beyond whether to build or buy. In fact, since the costs in time and money

are oftentimes prohibitive for internal development, the decision to build is typically the easiest decision to make for a large health information system such as the EHR. Coupled with the Meaningful Use EHR Incentive Program, adoption of at least basic EHRs in the United States is very high, exceeding 90% (see ■ Figs. 6.2, 6.3 and 6.4).¹

Once a decision to purchase a commercial system is made, the next decision is what system to purchase. There is a wide variation in the functionalities between different EHR systems, even though certification efforts have defined basic functions that each system should have (see ■ Fig. 6.3). Even systems with the same certified functions may approach the functions so differently that some implementations will be incongruent to an organization.² Key factors an organization should consider when choosing a system include (a) the core functionality of the software, including integration with other systems, (b) total system cost, (c) the service experience of other customers, and (d) the system’s certification status. Some organizations have performed systematic reviews of different commercial software offerings that can be a helpful start to identify possible vendors and understand variations between systems. For example, KLAS Research publishes periodic assessments of both software functions and vendor performance that can be used to identify potential software products. However, since systems are complex, it is important to meet with and discuss experiences with actual organizations that have used the software. This is typically done through site visits to existing customer organizations. It is also common for organizations to make a broad request of vendors for proposals to address a specific software need, especially

-
- 1 ▶ <https://v.healthit.gov/quickstats/pages/physician-ehr-adoption-trends.php> and ▶ <https://dashboard.healthit.gov/quickstats/pages/FIG-Hospital-EHR-Adoption.php>, from ▶ <https://dashboard.healthit.gov/quickstats/quickstats.php>
 - 2 ▶ <https://dashboard.healthit.gov/quickstats/pages/FIG-Vendors-of-EHRs-to-Participating-Hospitals.php> (last accessed June 3, 2020).

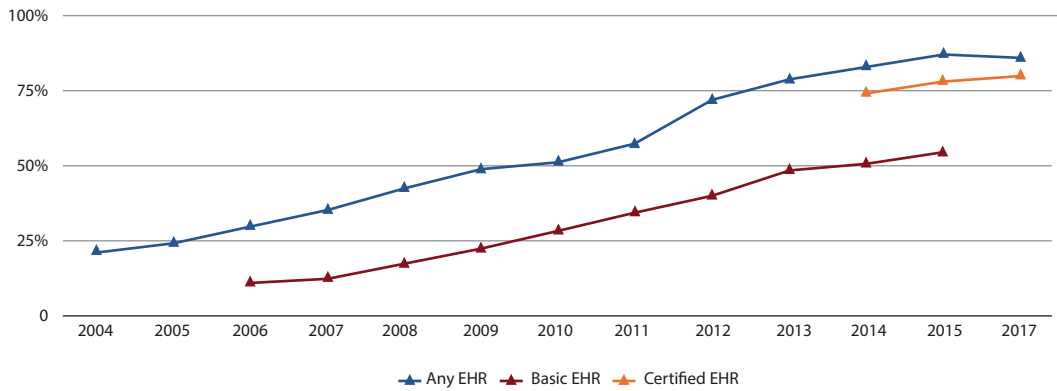


Fig. 6.2 Office-based physician EHR adoption, from ONC (2019a)

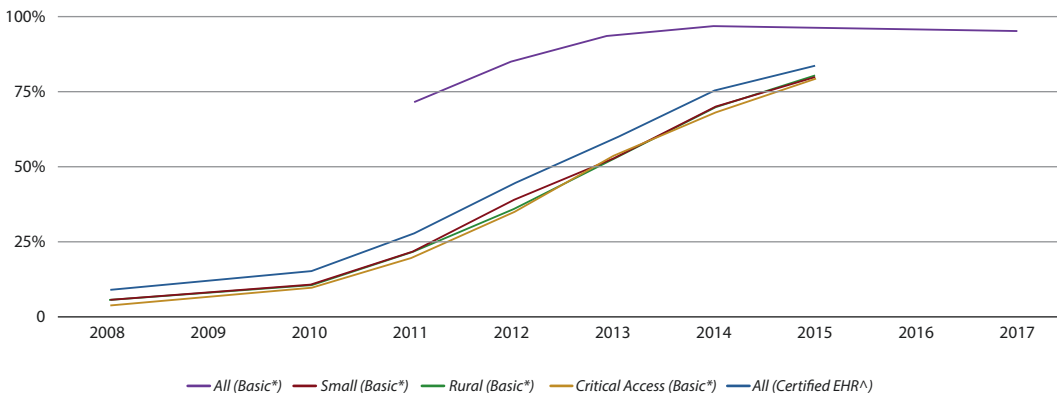


Fig. 6.3 Non-federal acute care hospital EHR adoption, from ONC (2017b)

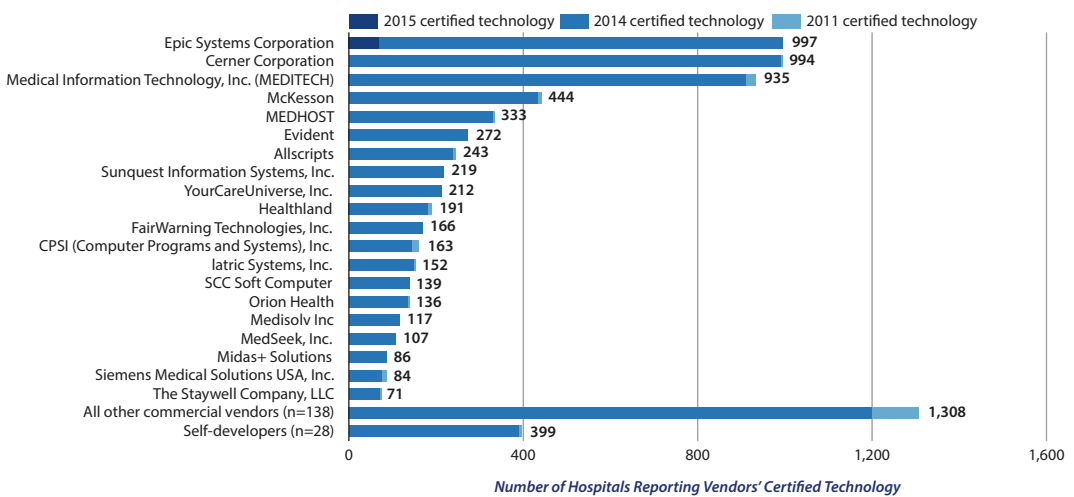


Fig. 6.4 Health IT developers product use by hospitals participating in the Medicare EHR, from ONC (2017a)

when the needs are not standard components of EHR software.

After a commercial product is selected, an organization must then choose how extensive the software will be. EHR companies typically have a core EHR system, with additional modules that have either been developed or acquired and integrated into their system. The set of modules used by each institution varies. One organization may use the core EHR system and accompanying modules for certain specialties, such as internal medicine and family practice, while choosing to purchase separate best-of-breed software for other specialties, like obstetrics/gynecology and emergency medicine, even when the core EHR vendor has functional modules for those areas. Another organization may choose to purchase and implement all specialty systems offered from the core EHR vendor, and only purchase other software if a similar module is not available from the vendor. These decisions also must be made for all ancillary systems, including laboratory, pharmacy, radiology, etc. This is both a pre-implementation decision and a long-term strategy. Once the EHR is implemented, many specialties that were not included in the initial implementation plan may request software and data integration, depending on the success of the EHR implementation.

For organizations that choose components of multiple vendor offerings to any degree, they will need to address how to integrate the components together to minimize disruption to the users' workflow. There are various strategies that can be pursued to integrate modules, either at the level of user context (user authentication credentials are maintained), the level of the application view (one application is viewable as a component within another application), or at the level of data sharing (data are exchanged between the applications). If components are not integrated, a user must access each application separately, by opening the software application, logging in to each separately, and selecting the patient within each. When data are integrated at the user context, a user moves between both applications, but the user and patient context are shared. This “single sign-

on” approach alleviates one of the main barriers to the user, by facilitating the login and patient selection, while retaining all the functionality of each system.

A deeper level of integration is at the application view. In this case, a primary application provides a portal that views another application; the second application shares user and patient context and is accessible through the user's main workflow system. The second application may use data from the primary application and/or other data sources. Rapidly expanding in adoption for this type of integration is the HL7 Substitutable Medical Applications and Reusable Technologies (SMART) framework, in particular in combination with HL7 Fast Healthcare Interoperability Resources (FHIR) for data interchange. This approach, known as SMART on FHIR, is enabling an ecosystem wherein applications developed by health care organizations or third-party vendors can be seamlessly integrated within the EHR (Kawamoto et al. 2019; Mandel et al. 2016).

The deepest level of integration is where 1 data elements from one system are also stored in the other system. With this approach, one system is determined to be the main repository, and data from the other systems are automatically stored into the repository. This approach has the advantage of the most complete use of data, e.g., decision support logic can use data from multiple systems, which can be more accurate. The disadvantage is that the integration can be expensive, requiring new interfaces for each integrated system.

Another and often overlooked consideration of EHR software modules is data analytics capabilities, usually discussed in conjunction with a data warehouse. EHR systems generally include reporting functionality, where specific reports can be configured to summarize and display data stored in the system. However, these systems often do not facilitate ad hoc data extracts that are commonly needed for more complicated data analysis. Additionally, if modules from multiple software vendors are used, the data reporting functions will be limited according to how well data are integrated. One typical approach

is to use a separate data warehouse and analysis system, with functions to create ad hoc reports, that can combine data from multiple systems. Data integration with warehouses is less expensive than with repositories, because the data do not need to be synchronized. Instead, data can be extracted in batches from source systems, transformed to the warehouse data model, and then loaded into the warehouse at periodic intervals. The greatest cost of integration is the data transformation, but this transformation is similar to what is required when receiving data through a real-time interface.

The **Meaningful Use** program, which has evolved to become the **Promoting Interoperability** program, has greatly influenced the systems that are installed by an institution. Initially, the ONC created a list of important EHR functions. They also created a requirement that hospitals and physician practices use a “certified” system – i.e., one that has demonstrated it provides those functions – to receive the incentives, and other criteria that the functions must be used in clinical care (Washington et al. 2017). As a result, health care organizations rapidly implemented EHRs that were certified and best fulfilled regulatory requirements.

6.3.3.2 Case Studies of EHR Adoption

Consider the following case studies of institutions adopting EHR systems. All examples are fictional, but reflect the complexity of the issues with EHR software.

Best-Care Medical Center had been using information systems for many years, dating back to when some researchers in the cardiology department built a small system to integrate data from the purchased laboratory and pharmacy information systems. Eventually, the infection control group for the hospital began using the system, and contributed efforts to expand its functionality. Other departments began developing decision support rules, and the system continued to grow. Eventually, the institution made a commitment to redevelop the infrastructure to support a much larger group of users and functions, and named it A-Chart. Satisfaction with the system was high where it had been initially developed, and with other

related specialties. However, over time there was disproportionate development in these areas, and clinicians in other specialties complained about the rudimentary functionality, especially when compared to existing vendor systems for their specialty. As a result, the organization decided to purchase a new vendor system. This made the other specialties happy, but was a big concern to the groups that had been using A-Chart for years. These clinicians feared that they would have to reconfigure their complicated decision support rules with a new system, or worse, that functionality would no longer be supported. To alleviate concerns, representatives from each department were asked to participate in both drafting a Request for Proposals and then reviewing the proposals from four different vendors. Many clinicians liked System X, but in the end the hospital chose System Y, which seemed to have most of the same capabilities but was perceived to be more affordable than System X. However, System Y did not include a laboratory system, so the medical center purchased a separate laboratory system and built interfaces to connect it with the core EHR.

Patients’ Choice is an integrated delivery system with a long history of EHR use. Years ago, it existed as two separate systems of hospitals and clinics. Shortly before the merger of these systems, the hospitals and clinics purchased separate EHRs, InPatSys and CliniCare. At the time of the merger, the institution felt that each would be best served by a best-of-breed system, to support the different workflows, and there was no single system that both sides of the organization could agree to use. Years later, as Patients’ Choice began to integrate care between the hospitals and clinics, the clinicians and administrators became increasingly frustrated at how different the InPatSys and CliniCare systems were, and that they had to use two separate systems to care for the same patients. A team was formed to evaluate the options, and the CliniCare system was eventually replaced by OutPatSys, the outpatient version of InPatSys. To prevent losing data as they moved from one system to the other, the Patients’ Choice IT department prepared the OutPatSys system by loading existing laboratory results and vital sign measurements from

CliniCare. A SMART on FHIR application provided an integrated view of historical CliniCare data inside the OutPatSys system.

Hometown Community Hospital historically used various niche information systems, but no EHR. With the availability of Meaningful Use incentives, the hospital decided to acquire a commercial EHR. A leadership team visited six hospitals to investigate how various EHRs were used. Ultimately, the hospital made the decision to purchase eCompuChart, because it was highly rated and seemed best adapted to their needs. Hometown hired a new Chief Information Officer who had recently implemented eCompuChart at a community hospital in a neighboring state. They also promoted Dr. Jones, who had recently moved from another hospital that had also used eCompuChart, to Chief Medical Information Officer (CMIO). The CIO and CMIO negotiated a contract with DigiHealth, a consulting company with experience in implementing EHRs, to plan and coordinate the implementation. Among other recommendations from DigiHealth, most existing systems were replaced with modules from eCompuChart to simplify maintenance.

In practice, organizations may not adopt a complete “build” or a complete “buy” strategy. EHR vendors have advanced considerably in their ability to create systems that meet common needs in health care. Still, no system exists to date that can fully address all information needs for an organization, in part because the information needs expand as more data and new technologies become available. Additionally, EHR strategies become malleable over time, as commercial software capabilities increase and data become more consistent. As indicated through some of the examples above, organizational strategies may change over time to adapt to these capabilities and needs. Expanding options for health care organizations is the emergence of the notion of the EHR as a platform, where HL7 FHIR data interfaces can be used to read data from, and in some cases write data back to the EHR; HL7 SMART is available to integrate external applications into the native EHR user interface; and more recently, a framework known as HL7 CDS Hooks is

available to interface externally developed alerts and reminders into the EHR.³

One consideration that is not always stated in the software selection process, but is significant in its influence over the decision, is how an organization will pay for the application. In organizations where software purchases are requested from the information technology department and budget, overall maintenance costs are considered more prominently, and software that integrates with and is a component of the overall EHR vendor offering is often selected. However, if a clinical department has direct control over their spending for the software, functionality may become a more focal concern. An additional case study illustrates this situation.

Downtown Hospital recently decided to purchase eCompuChart as the centerpiece of its overall clinical information system strategy. eCompuChart has award-winning modules for the emergency department and intensive care units. However, there are strong complaints about its capabilities for labor and delivery management and radiology. After considering capabilities of best-of-breed options and their ability to integrate with eCompuChart, Downtown Hospital eventually made a split decision. The labor and delivery module for eCompuChart was purchased because other systems with more elaborate functionality could not integrate data as well with the overall EHR. On the other hand, a separate best-of-breed system was purchased for radiology, because interfaces between the systems were seen as an acceptable solution for integrating data.

6.3.3.3 Enhancing Acquired Software

Although most institutions will choose to acquire a system rather than building it from scratch, software engineering is still required to make the systems function effectively. This involves more than just installing and configuring the software to the local environment. There is still a significant need for software development in implementing COTS, because

3 ▶ <https://cbs-hooks.org/> (last accessed June 3, 2020).

(1) applications must be integrated with existing systems, and (2) leading healthcare institutions are increasingly developing custom applications, such as SMART on FHIR applications, that supplement commercial systems.

6.3.3.4 Integration with Existing Systems

In all but the most basic health care information technology environments, multiple software applications are used for treatment, payment, and operations purposes. A partial list of applications that might be used in a hospital environment is shown in [Table 6.1](#).

To facilitate the sharing of information among various software applications, standards have emerged for exchanging messages and defining clinical terminology (see [Chap. 8](#)). Message exchange between different software applications enables the following scenario:

1. A patient is admitted to the hospital. A registration clerk uses the bed management system to assign the patient's location and attending physician of record.

Table 6.1 Partial list of software applications that may be used in a hospital setting

System	Primary Users
Inpatient EHR (Results Review, Order Entry, Documentation)	Physicians, nurses, allied health professionals
Pharmacy Information System	Pharmacists, pharmacy technicians
Laboratory Information System	Laboratory technicians, phlebotomists
Radiology Information System	Radiologists, radiology technicians
Pathology Information System	Pathologists
Registration/Bed Management	Registration staff
Hospital Billing System	Medical coders
Professional Services Billing System	Physicians, medical coders

2. The physician orders a set of routine blood tests for the patient in the inpatient EHR computerized order entry module.
3. The request for blood work is sent electronically to the laboratory information system, where the blood specimen is matched to the patient using a barcode.
4. The results of the laboratory tests are sent to the results review module of the EHR

Message exchange is an effective means of integrating disparate software applications in healthcare when the users rely primarily on a single “workflow system” (e.g., a physician uses the inpatient EHR and a laboratory technician uses the LIS). Because message exchange is handled by a sophisticated “interface engine” (see [Chap. 8](#)), little software development, in the traditional sense, is typically required. When a user accesses multiple workflow systems to perform a task, message exchange may not be sufficient and a deeper level of integration may be required. For example, consider the following addition to the previously described scenario:

5. The physician reviews the patient's blood work and notes that the patient may be suffering from renal insufficiency as evidenced by his elevated creatinine level.
6. The physician would like to review a trend of the patient's creatinine over the past 3 years. Because the hospital installed their commercial EHR less than a year ago, data from prior to that time are available in a legacy results review system that was developed locally. The physician logs into the legacy application (entering her username and password), searches for the correct patient, and reviews the patient's creatinine history.

While it may seem preferable in this scenario to load all data from the legacy system into the new EHR, commercial applications may not support importing such data for various reasons. To simplify and improve the user experience for reviewing information from a legacy application within a commercial EHR, one group of informaticians created the custom application shown in [Fig. 6.5](#). The application is accessed by clicking a link

Main Lab Summary

Basic Metabolic

One Week
One Month
Six Months
One Year
Two Years
Five Years
All Time

	Na	K	Cl	HCO3	BUN	Creat	Gluc	Ca	Mg	Phos	Urate	iCa
29Feb12 16:31	-	-	-	-	-	-	99	-	-	-	-	-
28Feb12 11:22	Tes	4.2	-	-	-	-	-	-	-	-	-	<0.25
28Feb12 10:40	154	4.0	110	21	20	0.90	100	8.0	-	-	-	-
28Feb12 10:37	150	4.0	110	20	25	0.80	110	8.0	-	-	-	-
27Feb12 15:49	-	-	-	-	-	-	419	-	-	-	-	-
27Feb12 14:33	-	-	-	-	-	-	158	-	-	-	-	-
27Feb12 10:09	155	8.3	115	22	20	0.90	100	8.5	-	-	-	-
28Feb12 15:59	-	-	-	-	-	-	106	-	-	-	-	-
24Feb12 19:28	-	-	-	-	-	-	143	-	-	-	-	-
24Feb12 03:06	-	-	-	-	-	-	87	-	-	-	-	-

• **Fig. 6.5** Example screen from a custom lab summary display application integrated into a commercial EHR. The application shows a longitudinal view of laboratory results that can span multiple patient encounters

SANDIEGO, CARMEN V - iCharge

Billing History Encounter Form **Diagnosis** Visit Charges Procedure Charges About

Encounter Form [Select Encounter](#)

Day Date Linked Note [Select from 1 available](#)
 MON 19 Mar 2012 [Surgery Attending Free Text Note 19 Mar 2012 1055](#)

Billing Provider: Division:
 Performing Provider: Billing Area:
 Referring Provider: Location:
 Compliance Code: [Compliance Code Definitions](#)

Primary	#	ICD Code	Diagnosis
Yes	1	250.00	Diabetes mellitus without mention of complication, type II or unspecified type, not stated as uncontrolled
	2	291.81	Alcohol withdrawal
	3	428.31	Acute diastolic heart failure

CPT Code	Modifiers	Units	Charge	Linked Dxs
99232	25	1.0	SBSQ HOSP CARE PR D 25 MIN	1, 2, 3

2 Eclipse Health Issues not mapped

• **Fig. 6.6** Example screen from a custom billing application integrated into a commercial EHR. This replaced a separate application that was not integrated into the clinicians' workflow

within the commercial EHR and does not require login or patient look-up.

In an example of a more sophisticated level of “workflow integration” is shown in • Fig. 6.6. In this example, informaticians developed a custom billing application within

an inpatient commercial EHR. Users of the application were part of a physician practice that used a different outpatient EHR with a professional billing module with which they were already familiar. When the physicians in the practice rounded on their patients who


were admitted to the hospital, they documented their work by writing notes within the inpatient EHR, and then used their outpatient EHR to submit their professional service charges. This practice not only required a separate login to submit a bill, but also required duplicate patient lists to be maintained in each application, as well as a duplicate problem list for each patient to be managed in each application. The integrated charge application was accessed from the inpatient EHR but provided the same look-and-feel as the outpatient EHR billing module. Charges were submitted through the outpatient EHR infrastructure and would appear as normal charges in the outpatient system, with the substantial improvement of displaying the information (note name, author, and time) for the documentation that supported the charge.


6.3.3.5 Development of Custom Applications that Supplement or Enhance Commercial Systems

Commercial EHRs frequently provide customers with the ability to develop custom software modules. Some EHRs provide a flexible clinical decision support infrastructure that allows customers to develop modules that execute medical logic to generate alerts, reminders, corollary orders, and so on. Vendors may also provide customers with tools to access the EHR database, which allows development of stand-alone applications that make use of EHR data. Additionally, vendors may foster development of custom user interfaces within the EHR by providing an application programming interface through which developers can obtain information on user and patient context.

The ability to provide patient-specific clinical decision support is one of the key benefits of EHRs. Many commercial EHRs either directly support or have been influenced by the **Arden Syntax for Medical Logic Modules** (Pryor and Hripcsak 1993). The Arden Syntax is part of the HL7 family of standards. It encodes medical knowledge as **Medical Logic**

Modules (MLMs), which can be triggered by various events within the EHR (e.g., the placing of a medication order) and execute serially as a sequence of instructions to access and manipulate data and generate output. MLMs have been used to generate clinical alerts and reminders, to screen for eligibility in clinical research studies, to perform quality assurance functions, and to provide administrative support (Dupuits 1994; Jenders 2008; Jenders and Shah 2001; Ohno-Machado et al. 1999). Although one goal of the Arden Syntax was to make knowledge portable, MLMs developed for one environment are not easily transferable to another. Developers of clinical decision support logic require skills in both computer programming as well as medical knowledge representation.

An example of a standalone, locally developed software application that relies on EHR data is shown in  Fig. 6.7. The Web-based application, EpiPortal™, provides a comprehensive, electronic hospital epidemiology decision support system. The application can be accessed from a Web browser or directly from within the EHR. It relies on EHR data such as microbiology results, clinician orders, and bed tracking information to provide users with timely information related to infection control and prevention.

In some cases, it is desirable to develop custom applications to address specific clinical needs that are not met by a commercial EHR. For example, most commercial EHRs lack dedicated tools to support patient hand-off activities. For hospitalized patients, hand-offs between providers affect continuity of care and increase the risk of medical errors. Informaticians at one academic medical center developed a collaborative application supporting patient handoff that is fully integrated with a commercial EHR (Fred et al. 2009). An example screen from the application is shown in  Fig. 6.8. The application creates user-customizable printed reports with automatic inclusion of patient allergies, active medications, 24-hour vital signs, recent common laboratory test results, isolation requirements, code status, and other EHR data.

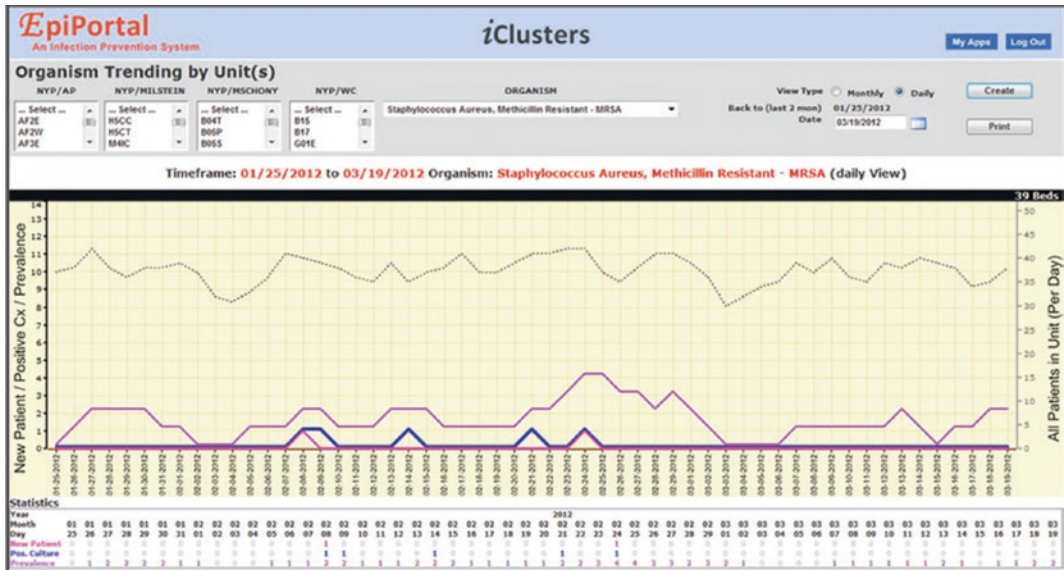


Fig. 6.7 Example screen from a standalone, software application that relies on EHR data to provide a comprehensive, electronic hospital epidemiology decision support system. (Reused with permission. Copyright 2012 The New York and Presbyterian Hospital and Columbia University – All rights reserved. EpiPortal is a trademark of The New York and Presbyterian Hospital.)

right 2012 The New York and Presbyterian Hospital and Columbia University – All rights reserved. EpiPortal is a trademark of The New York and Presbyterian Hospital.)

6.4 Emerging Influences and Issues

Several trends in software engineering are beginning to significantly influence biomedical information systems. While many of the trends may not be considered new to software engineering in general, they are more novel to the biomedical environment because of the less rapid and less broad adoption of information technology in this field. One area in particular that has received growing attention is **service oriented architectures (SOA)**. Sometimes called “software as a service”, SOA is a software design framework that allows specific processing or information functions (services) to run on an independent computing platform that can be called by simple messages from another computer application. For example, an EHR application might have native functionality to maintain a patient’s medication list, but might call a drug-drug interaction program running on a third party system to check the patient’s medications for potential interactions. This allows the EHR provider to off-load developing this

functionality, while the drug-drug interaction service provider can concentrate efforts on this focused task, and in particular on ensuring that the drug interaction database is kept up-to-date for all users of the service. Since the service is independent of any EHR application, many different EHR providers can call the same service, as can other applications such as patient health record (PHR) applications that are focused on consumer functionality. SOA might also be grouped with the more recently computer phrase “cloud computing”, which includes providing functional services to other applications, but also encompasses running entire applications and storing data in offsite or disconnected locations. A good example of SOA is the HL7 CDS Hooks standard, which specifies how EHR systems can interface with external clinical decision support services to provide point-of-care alerts and reminders to clinical end users.

Another emerging trend, discussed earlier, is the notion of the **EHR as a platform** for third-party applications and services that interface with, and add value to, the EHR. Central to this approach is HL7 FHIR

MICU-1234-01 **LASTNAME, FIRSTNAME (123 45 67)** - 66y M / 72.5 kg - **FULL CODE**

Team: Unassigned / Attg: Xtest, Doctor Adm:00-00-2009 / LOS:2d

<p>Patient Summary Working Dx: Upper GI bleed 66 yo man with only known history long term heavy etoh abuse...</p> <p>Day 1: EGD showing gastric ulcer with visible vessel and gastric varices...</p> <p>Day 0: in er had tachy cardia got 2L but still orthostatic so sent to micu for observation.</p> <p>Notes/Comments Attending: Xtest Care Category: Ward (On Service) Contact Info: only phone is his cell phone Daughter has it (646-555-5555)</p> <p>abx: metronidazole 500mg iv q8h (00/00-)</p> <p>past abx: ceftriaxone 00/00-00/00</p> <p>Tubes/Lines/Drains:R> iv access : cortis (00/00) 2 18 gauge</p>	<p>Allergies No Known Allergies</p> <p>Meds: Drips Esomeprazole DRIP</p> <p>Meds: Standing Metronidazole Inj 500mg IVPB q8hr Phytonadione Inj 1MG IV q24h Calcium Gluconate inj 2G IV Once Folic Acid Inj 1mg IVPB q24h Pneumococcal 23-Valent Vacc (Pneumovax) 0.5ml IM Thiamine HCl Inj 100MG IV q24h Potassium Phosphate Inj 15mmol IVPB Once</p> <p>I&O</p> <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th rowspan="2">Item</th> <th colspan="4">7A (00:00) - 7A (00:00) since</th> </tr> <tr> <th>12h</th> <th>12h</th> <th>24h</th> <th>7A</th> </tr> </thead> <tbody> <tr> <td>D5W</td> <td>100</td> <td>350</td> <td>450</td> <td>500</td> </tr> <tr> <td>Esomeprazole DRIP</td> <td>120</td> <td>120</td> <td>240</td> <td>100</td> </tr> <tr> <td>Normal Saline (...)</td> <td>110</td> <td>90</td> <td>200</td> <td>0</td> </tr> <tr> <td>Octreotide DRIP</td> <td>21</td> <td>0</td> <td>21</td> <td>0</td> </tr> <tr> <td>Total In</td> <td>351</td> <td>560</td> <td>911</td> <td>600</td> </tr> <tr> <td>Urine: Voided</td> <td>2030</td> <td>470</td> <td>2500</td> <td>500</td> </tr> <tr> <td>Total Out</td> <td>2030</td> <td>470</td> <td>2500</td> <td>500</td> </tr> <tr> <td>TOTAL NET</td> <td>-1679</td> <td>90</td> <td>-1589</td> <td>100</td> </tr> </tbody> </table> <p>Vitals (last 2-4h) T_c : 36.2 (T_{max}: 37.2 @ 00/00 19:45) HR: 81 (66 - 98) BP: 152/97 (99/57 - 152/97) RR: 19 (16 - 31) SpO₂: 100% (96 - 100)</p> <p>Labs (last 2-4h)</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: center;">6.8</td> <td style="text-align: center;">9.3</td> <td style="text-align: center;">119</td> <td style="text-align: center;">00/00</td> </tr> <tr> <td style="text-align: center;">26.9</td> <td style="text-align: center;">26.9</td> <td style="text-align: center;">09:20</td> <td></td> </tr> <tr> <td style="text-align: center;">141</td> <td style="text-align: center;">111</td> <td style="text-align: center;">13</td> <td style="text-align: center;">00/00</td> </tr> <tr> <td style="text-align: center;">3.3</td> <td style="text-align: center;">22</td> <td style="text-align: center;">0.7</td> <td style="text-align: center;">03:00</td> </tr> <tr> <td style="text-align: center;">7.8</td> <td style="text-align: center;">9.1</td> <td style="text-align: center;">114</td> <td style="text-align: center;">00/00</td> </tr> <tr> <td style="text-align: center;">26.8</td> <td style="text-align: center;">26.8</td> <td style="text-align: center;">03:00</td> <td></td> </tr> </table>	Item	7A (00:00) - 7A (00:00) since				12h	12h	24h	7A	D5W	100	350	450	500	Esomeprazole DRIP	120	120	240	100	Normal Saline (...)	110	90	200	0	Octreotide DRIP	21	0	21	0	Total In	351	560	911	600	Urine: Voided	2030	470	2500	500	Total Out	2030	470	2500	500	TOTAL NET	-1679	90	-1589	100	6.8	9.3	119	00/00	26.9	26.9	09:20		141	111	13	00/00	3.3	22	0.7	03:00	7.8	9.1	114	00/00	26.8	26.8	03:00		<p>Primary Team To Do <input type="checkbox"/> has protein gap send hiv/hepc/mm <input type="checkbox"/> sent hep w/u <input type="checkbox"/> watch for signs of withdrawal <input type="checkbox"/> f/u fibrinogen and will consider cryo if less than 100 <input type="checkbox"/> f/u beta 2 microglobulin <input type="checkbox"/> TTE <input type="checkbox"/> troponin trend</p> <p>Coverage To Do <input type="checkbox"/> f/u u/s <input type="checkbox"/> Hpylori IgG ordered, if positive --> treat</p>
Item	7A (00:00) - 7A (00:00) since																																																																										
	12h	12h	24h	7A																																																																							
D5W	100	350	450	500																																																																							
Esomeprazole DRIP	120	120	240	100																																																																							
Normal Saline (...)	110	90	200	0																																																																							
Octreotide DRIP	21	0	21	0																																																																							
Total In	351	560	911	600																																																																							
Urine: Voided	2030	470	2500	500																																																																							
Total Out	2030	470	2500	500																																																																							
TOTAL NET	-1679	90	-1589	100																																																																							
6.8	9.3	119	00/00																																																																								
26.9	26.9	09:20																																																																									
141	111	13	00/00																																																																								
3.3	22	0.7	03:00																																																																								
7.8	9.1	114	00/00																																																																								
26.8	26.8	03:00																																																																									

Fig. 6.8 Example screen from a custom patient handoff application integrated into a commercial EHR. The application creates user-customizable printed

reports with automatic inclusion of patient allergies, active medications, 24-hour vital signs, recent common laboratory test results, isolation requirements, code status, and other EHR data

application programming interface (API), which uses modern Internet technologies and approaches for data exchange, as well as HL7 SMART for application integration and HL7 CDS Hooks for integrating decision support services. While this notion of EHR as a platform is still in its early stages and still maturing, many EHR vendors are strongly supportive of this type of an ecosystem, and promising examples are emerging of how these technologies can be used to deliver value to health care organizations in an EHR-agnostic manner. We anticipate that this approach to health information systems, wherein core EHR systems are augmented by

third-party applications and services, will play an important role in the health IT ecosystem in the years to come.

Another important consideration in clinical information systems is infrastructure to support data sharing, such as through a **health information exchange (HIE)**. HIE infrastructure allows organizations to share information about patients through a common electronic framework. Robust HIE capabilities, which are now being implemented in commercial EHR systems, make it much more efficient to share patient information between organizations versus creating point-to-point interfaces between all the clinical information

systems a particular provider might need to communicate with. Effective sharing of information is predicated on the use standard message formats and terminologies (see ► Chap. 8), or the use of a shared EHR vendor. Where HIE functionality does not exist or is declining (Adler-Milstein et al. 2016), sharing can be coordinated directly between organizations with incentives for sharing, such as in an ambulatory care network (ACN). As health care in the United States increasingly shifts to a payment model that rewards value over volume, data sharing capabilities and the capacity for population-level analytics and health management will become increasingly critical. Moreover, there are also emerging efforts to scale health information exchange to a national scale, and to facilitate patient access to their information using APIs (ONC 2019b).

Software engineering is an ever-evolving discipline, and new ideas are emerging rapidly in this field. It is less than 30 years since the first graphical browser was used to access the World Wide Web, but today Web-based applications are the standard. Access to information through search engines has changed the way that people find and evaluate information. Social networking applications have altered our views on privacy and personal interaction. All of these developments have shaped the development of healthcare software, too. Today it is unimaginable that an EHR would not support a Web-based patient portal. Clinicians and consumers use the Web to search for health-related information in growing numbers and with growing expectations. It is not atypical for patients to discuss health issues in online forums and share intimate details on patient networking sites.

Another development that is impacting virtually all industries, including health care, is **advanced analytics**. Coupled with the rapid increase in the adoption of EHR systems, health care represents a golden opportunity for leveraging powerful computing approaches with large data sets to identify and apply new insights. For example, deep learning techniques can be applied to EHR data to predict important outcomes such as in-hospital mortality. If coupled

with user-centered, workflow-integrated interventions, such insights have the potential to improve clinical decision making and enhance patient care.

6.5 Summary

The goal of software engineering in health care is to create a system that facilitates delivery of care. Much has changed in the past decade with EHRs, and today most institutions will purchase rather than build an EHR. But engineering these systems to facilitate care is still challenging, and following appropriate software development practices is increasingly important. The success of a system depends on interaction among designers of healthcare software applications and those that use the systems. Communication among the participants is very difficult when it comes to commercial applications. Informaticians have an important role to play in bridging the gaps among designers and users that result from the wide variety in background, education, experience, and styles of interaction. They can improve the process of software development by specifying accurately and realistically the need for a system and of designing workable solutions to satisfy those needs.

Suggested Reading

Carter, J. H. (2008). *Electronic health records* (2nd ed.). Philadelphia: ACP Press. Written by a clinician and for clinicians, this is a practical guide for the planning, selection, and implementation of an electronic health record. It first describes the basic infrastructure of an EHR, and then how they can be used effectively in health care. The second half of the book is written more as a workbook for someone participating in the selection and implementation of an EHR.

KLAS Reports. <http://www.klasresearch.com/reports>. These reports are necessary tools for a project manager who needs to know the latest industry and customer information about vendor health information technology products. The reports include information on functionality available from vendors as well as customer

opinions about how vendors are meeting the needs of organizations and whose products are the best in a particular user environment.

McConnell, S. (1996). *Rapid development: Taming wild software schedules*. Redmond: Microsoft Press. For those who would like a deeper understanding of software development and project methodologies like Agile, this is an excellent source. It is targeted to code developers, system architects, and project managers.

President's Council of Advisors on Science and Technology (December 2010). Report to the President Realizing the Full Potential of Health Information Technology to Improve Healthcare for Americans: the Path Forward. <http://www.whitehouse.gov/sites/default/files/microsites/ostp/pcast-health-it-report.pdf>. This PCAST report focuses on what changes could be made in the field of electronic health records to make them more useful and transformational in the future. It gives a good summary of the current state of EHRs in general, and compares the barriers to those faced in adopting information technology in other fields. Time will tell if the suggestions really become the solution.

Stead, W. W., & Lin, H. S. (Eds.). (2009). *Computational technology for effective health care: immediate steps and strategic directions*. Washington, DC: National Academies Press. This is a recent National Research Council report about the current state of health information technology and the vision of the Institute of Medicine about how such technology could be used. It can help give a good understanding of how health IT could be used in health care, especially to technology professionals without a health care background.

Tang, P. C. (2003). *Key capabilities of an electronic health record system*. Washington, DC: National Academies Press. This is a short, letter report from an Institute of Medicine committee that briefly describes the core functionalities of an electronic health record system. Much of the report is tables that list specific capabilities of EHRs in some core functional areas, and indicate their maturity in hospitals, ambulatory care, nursing homes, and personal health records.

Wager, K. A., Lee, F. W., & Glaser, J. P. (2017). *Health care information systems: a practical approach for health care management*. John

Wiley & Sons. This is a textbook giving a good overview of healthcare information systems, used in many academic courses on the subject. It reviews the different environmental factors and contexts that influence the health information landscape nationally, as well as giving guidance on implementation, management and evaluation of systems.

? Questions for Discussion

- Reread the hypothetical case study in ► Sect. 6.2.1.
 - What are three primary benefits of the software used in James's care?
 - How many different ways is James's information used to help manage his care?
 - Without the software and information, how might his care be different?
 - How has health care that you have experienced similar or different to this example?
- For what types of software development projects would an agile development approach be better than a waterfall approach? For what types of development would waterfall be preferred?
- What are reasons an institution would choose to develop software instead of purchase it from a vendor?
- How is would various stages in the software development life cycle be different when developing software versus configuring or adding enhancements to an existing software program?
- Reread the case studies in ► Sect. 6.3.3.2.
 - What are the benefits and advantages of the different approaches to development and acquisition among the scenarios?
 - What were the initial costs for each institution for the software? Where will most of the long-term costs be?
- In what ways might new trends in software (small "apps" that accomplish focused tasks) change long-term strategies for electronic health record architectures?

References

- Adler-Milstein, J., Lin, S. C., & Jha, A. K. (2016). The number of health information exchange efforts is declining, leaving the viability of broad clinical data exchange uncertain. *Health Affairs*, 35(7), 1278–1285.
- Bates, D. W. (2006). Invited commentary: The road to implementation of the electronic health record. *Proceedings (Baylor University. Medical Center)*, 19(4), 311–312.
- Bates, D. W., Leape, L. L., et al. (1998). Effect of computerized physician order entry and a team intervention on prevention of serious medication errors. *JAMA*, 280(15), 1311–1316.
- Beck, K., Beedle, M., et al. (2001). *Manifesto for agile software development*. 2012, from agilemanifesto.org.
- Blumenthal, D., & Tavenner, M. (2010). The “meaningful use” regulation for electronic health records. *The New England Journal of Medicine*, 363(6), 501–504.
- Campanella, P., Lovato, E., Marone, C., Fallacara, L., Mancuso, A., Ricciardi, W., & Specchia, M. L. (2016). The impact of electronic health records on healthcare quality: A systematic review and meta-analysis. *European Journal of Public Health*, 26(1), 60–64.
- Chaudhry, B., Wang, J., et al. (2006). Systematic review: Impact of health information technology on quality, efficiency, and costs of medical care. *Annals of Internal Medicine*, 144(10), 742–752.
- Dupuits, F. M. (1994). The use of the Arden Syntax for MLMs in HIOS+, a decision support system for general practitioners in The Netherlands. *Computers in Biology and Medicine*, 24(5), 405–410.
- Evans, R. S., Pestotnik, S. L., et al. (1998). A computer-assisted management program for antibiotics and other anti-infective agents. *The New England Journal of Medicine*, 338(4), 232–238.
- Fred, M., Vawdrey, D., et al. (2009). *Enhancing a commercial electronic health record with a novel patient handoff application*. Society of Hospital Medicine (SHM) Annual Meeting, Chicago, IL.
- Han, Y. Y., Carcillo, J. A., et al. (2005). Unexpected increased mortality after implementation of a commercially sold computerized physician order entry system. *Pediatrics*, 116(6), 1506–1512.
- Hripcsak, G., Kuperman, G. J., et al. (1998). Extracting findings from narrative reports: Software transferability and sources of physician disagreement. *Methods of Information in Medicine*, 37(1), 1–7.
- Hunt, D. L., Haynes, R. B., et al. (1998). Effects of computer-based clinical decision support systems on physician performance and patient outcomes: A systematic review. *JAMA*, 280(15), 1339–1346.
- Jenders, R. A. (2008). Suitability of the Arden Syntax for representation of quality indicators. *American Medical Informatics Association Annual Symposium Proceedings*, 991.
- Jenders, R. A., & Shah, A. (2001). Challenges in using the Arden Syntax for computer-based nosocomial infection surveillance. In *Proceedings of the AMIA Symposium* (pp. 289–293). American Medical Informatics Association.
- Jones, S. S., Rudin, R. S., Perry, T., & Shekelle, P. G. (2014). Health information technology: An updated systematic review with a focus on meaningful use. *Annals of Internal Medicine*, 160(1), 48–54.
- Kawamoto, K., Kukhareva, P., Shakib, J. H., Kramer, H., Rodriguez, S., Warner, P. B., Shields, D., Weir, C., Fiol, G. D., Taft, T., & Stipelman, C. H. (2019). Association of an electronic health record add-on app for neonatal bilirubin management with physician efficiency and care quality. *JAMA Network Open*, 2(11), e1915343–e1915343.
- Leavitt, M., & Gallagher, L. (2006). The EHR seal of approval: CCHIT introduces product certification to spur EHR adoption. *Journal of AHIMA*, 77(5), 26–30, quiz 33–24.
- Mandel, J. C., Kreda, D. A., Mandl, K. D., Kohane, I. S., & Ramoni, R. B. (2016). SMART on FHIR: A standards-based, interoperable apps platform for electronic health records. *Journal of the American Medical Informatics Association*, 23(5), 899–908.
- McConnell, S. (1996). *Rapid development: Taming wild software schedules/Steve McConnell*. Redmond, Wash: Microsoft Press.
- Office of the National Coordinator for Health Information Technology (ONC). (2017a). *Certified health IT developers and editions reported by hospitals participating in the medicare EHR incentive program, Health IT Quick-Stat #29*. dashboard.healthit.gov/quickstats/pages/FIG-Vendors-of-EHRs-to-Participating-Hospitals.php. July 2017.
- Office of the National Coordinator for Health Information Technology (ONC). (2017b). *Non-federal acute care hospital electronic health record adoption, Health IT Quick-Stat #47*. dashboard.healthit.gov/quickstats/pages/FIG-Hospital-EHR-Adoption.php. September 2017.
- Office of the National Coordinator for Health Information Technology (ONC). (2019a). *Office-based physician electronic health record adoption, Health IT Quick-Stat #50*. dashboard.healthit.gov/quickstats/pages/physician-ehr-adoption-trends.php. January 2019.
- Office of the National Coordinator for Health Information Technology (ONC). (2019b). *Trusted exchange framework and common agreement | HealthIT.gov*. (2019). Retrieved February 2020, from <https://www.healthit.gov/topic/interoperability/trusted-exchange-framework-and-common-agreement>
- Ohno-Machado, L., Wang, S. J., et al. (1999). Decision support for clinical trial eligibility determination in breast cancer. In *Proceedings of the AMIA symposium* (pp. 340–344). American Medical Informatics Association.
- Pryor, T. A., & Hripcsak, G. (1993). The Arden syntax for medical logic modules. *International Journal of Clinical Monitoring and Computing*, 10(4), 215–224.

- Shekelle, P. G., Morton, S. C., et al. (2006). Costs and benefits of health information technology. *Evid Rep Technol Assess (Full Rep)*, 132(April), 1–71.
- Smelcer, J., Miller-Jacobs, H., et al. (2009). Usability of electronic medical records. *Journal of Usability Studies*, 4(2), 70–84.
- Sullivan, T. (2017). Coast guard seeks new EHR vendor after failed epic implementation. *Healthcare IT news*. <https://www.healthcareitnews.com/news/coast-guard-seeks-new-ehr-vendor-after-failed-epic-implementation>.
- Thompson, C.B., Snyder-Halpern, R., & Stagers, N. (1999). Clinical informatics case studies: Analysis, processes, and techniques. *Computers in Nursing*, 17(5), 203–206.
- Washington, V., DeSalvo, K., et al. (2017). The HITECH era and the path forward. *The New England Journal of Medicine*, 377, 904–906.