Appendix B
General Concepts

To best evaluate the detailed assessments and high-level evaluations of individual FOSS EHR projects, it is helpful to review several general features that vary among projects and influence their strengths and limitations. These features include the projects' (1) organizational structures, (2) open-source licensing models, (3) technical architectures, and (4) software deployment mechanism(s). Readers familiar with these features may skip this appendix.

Open Source Licensing Models
There are more than 50 types of open source licenses in use today (as listed by the Open Source Initiative). However, only three of these are relevant to the FOSS EHR projects reviewed in this paper: (1) the GNU General Public License (GPL), (2) the GNU Lesser Public License (LGPL), (3) and the Mozilla Public License (MPL).

GPL: The GPL stipulates that anyone in possession of the licensed software may use, copy, edit, and redistribute it with only two restrictions: First, the software must be redistributed with its source code to ensure that it remains open; second, the software may only be redistributed under the GPL, to ensure that the existing rights to it are never curtailed by future recipients. These are the so-called copyleft provisions, because they prevent any individual who comes into possession of open-source software from appropriating exclusive rights to it or its derivative works (exactly the opposite of a copyright, which establishes and protects such proprietary rights). Other noteworthy features of the GPL are (1) it does not require that modifications to open-source software be published or shared, only that if they are, it be done under the terms described above, (2) it does not allow open source software code to be incorporated into any proprietary software, either via tight integration or loose coupling, and (3) it does allow open-source software or its derivative works to be sold for money, provided that the terms of the GPL are preserved in any distributed copies (i.e., the distributor may charge for the software, but must distribute it with its source code and cannot prohibit the purchaser from further distributing it free of charge).

LGPL: A weaker version of the GPL license that imposes most of the same copyleft restrictions on redistributed software as the GPL licenses. The LGPL is intended to be used primarily for libraries that offer encapsulated functionality. Changes to, and redistributions of, the library itself fall under the same copyleft protections as GPL software. Unlike GPL, however, LGPL allows other applications (both open source and proprietary) to link to or use an LGPL library without being considered a derivative work (and thereby no longer proprietary).

MPL: A weaker variety of copyleft licensing. Like GPL, MPL also requires any source code that is copied or changed to remain under the MPL distribution agreement. However, unlike GPL, the source code protected under MPL may be incorporated into software that is distributed under any other license (including a proprietary one). This allows recipients to develop proprietary products that contains MPL-covered code, as long as the MPL portion of the code is distributed under the original license agreement (i.e., with its source code).

In addition to these flavors of open-source licenses, many FOSS projects (including the EHR projects discussed here) also require that programmers or companies that contribute code to sign a contributor license agreement, which is another mechanism to protect the openness of the software. These include:

CLA: A CLA grants all copyright and patent licenses for the software to the nonprofit or for-profit project that is receiving the code, as well as to any subsequent recipients who may receive the code from that project. Having CLAs in place allows the project to legally defend the status of all its source code in the event of a dispute with the contributor or other parties at a later time.

Organizational and Management Structures

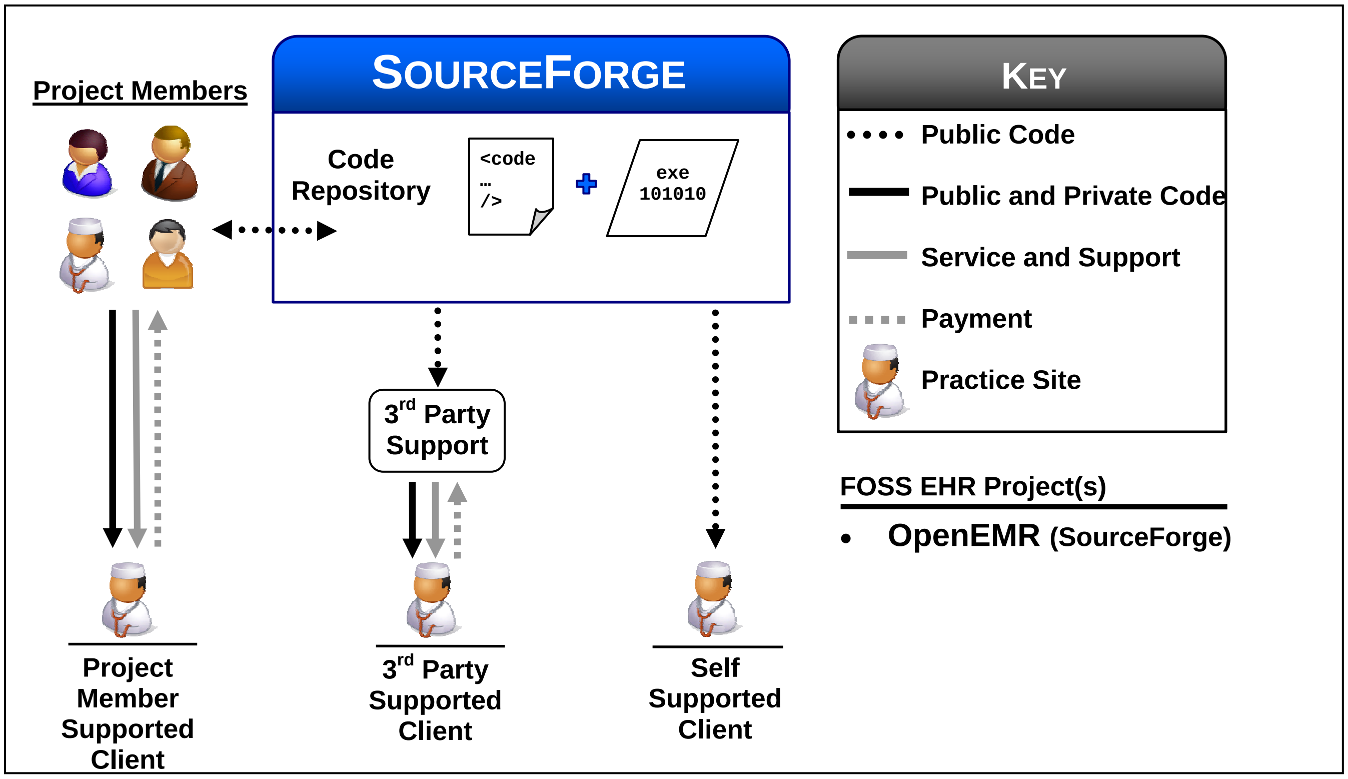
The size and stability of the software developer community can greatly influence the longevity and success of an open-source project. The organizational structure of the project, in turn, largely determines the stability of the developer community. There are currently three main organizational structures among the projects reviewed in detail for this analysis: (1) Community Organizations, (2) Commercial Organizations, and (3) Nonprofit Organizations.

Community Organizations

Projects operating under the community model are decentralized and loosely organized. They are typically led in an informal manner by one or two individuals and staffed by a group of volunteer project members. The primary organizing mechanism of the project is the source-code management system itself (for example, SourceForge) and the features it provides for the submission of code, downloading of code, and communication among programmers and testers. In this model, the project members who contribute code are typically (1) hobbyists who have a history with the project, (2) third-party support firms that whom have contracted to add features to the core application, or (3) clinicians at the practice sites that use the system. The development of features is decentralized and dictated exclusively by the requirements of those users who have the knowledge and/or resources to do the programming.

A diagram of a community organization support structure is presented in Figure 1.

Figure 1 - Community Organization Support Structure



Although the community model is arguably the purest form of open source software development, it is the exception among FOSS EHR projects, because it is relatively unstable. For example, the departure of a single key contributor from a community open source project can jeopardize the entire project.

Revenue Generation

There is no revenue-generation mechanism in place for a community FOSS EHR project. Project leaders and individual programmers volunteer their time to the development and maintenance of the software (although some contributors may be paid by their employers to do so). Although third-party support firms charge practice sites for installation, configuration, and maintenance services, these revenues accrues to the individual support firms, rather than to the project, per se.

Support and Service Options

Practice sites interested in implementing the FOSS EHR solution have three options. First, they may contract with a third-party support firm that is staffed by active project members familiar with the application. This firm can install and configure the system, implement data interfaces, and customize the software, as needed.

Alternatively, practice sites may contract with independent third-party support firms that have not participated in the software development for the FOSS EHR project, but have some knowledge of the system. Although these firms can also provide system installation and maintenance services, they may not be sufficiently familiar with the project code base to provide effective customization, configuration, and data-interfacing services.

Lastly, technically savvy practice sites may download the open source software and implement it themselves. This approach is typically feasible only if personnel at the practice site are comfortable with the operating system administration tasks necessary to install the product and provide the security and backup solutions necessary to entrust the system with patient data.

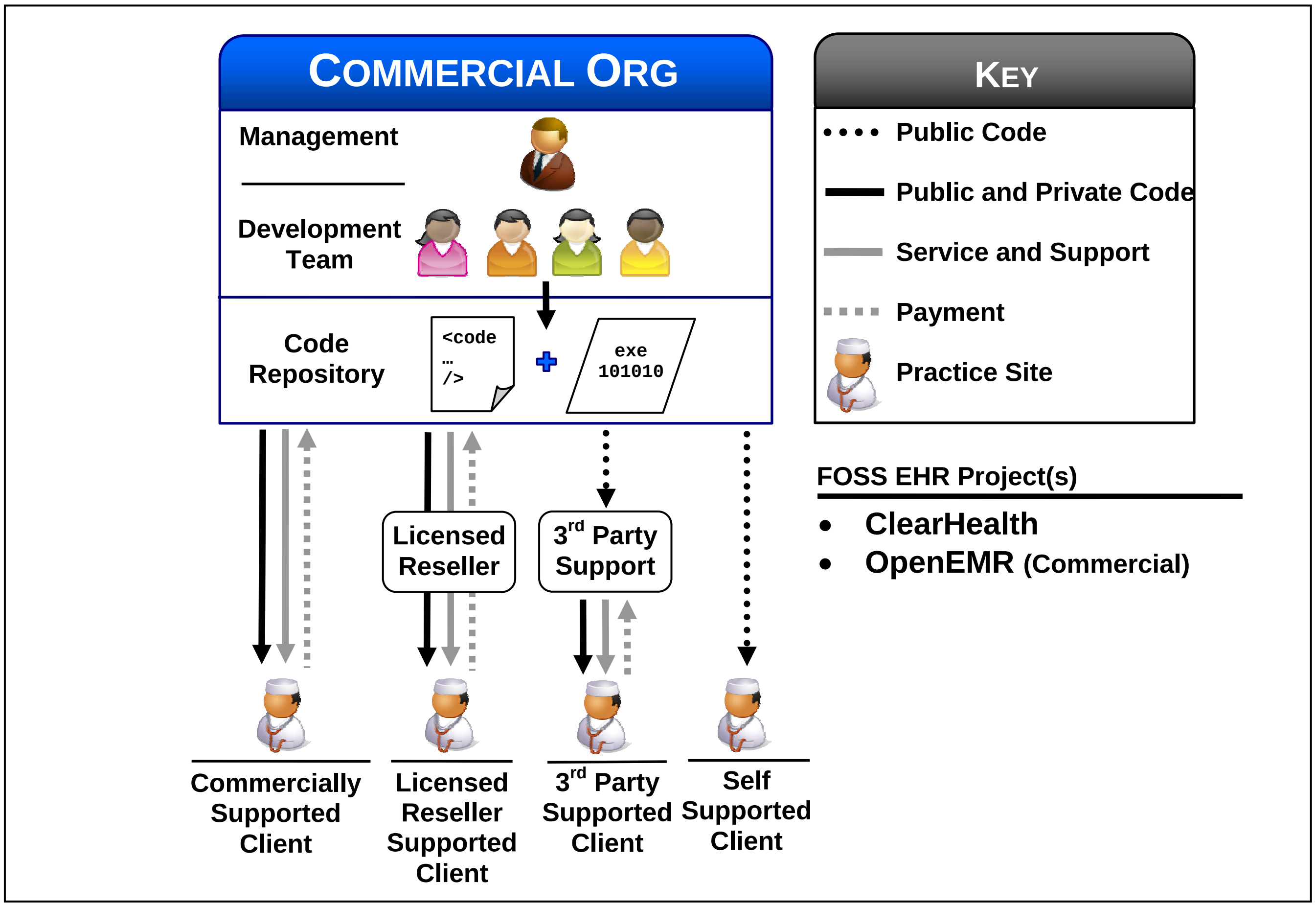
Source Code Management

The source code for community organization projects is maintained in a publicly accessible repository, such as SourceForge. Within these repositories, anyone is free to download the source code and install or modify it, subject to the limitations of their own expertise. To contribute new code to the repository, one must register as a member of the project. Registration is controlled by a project leader or leaders, who are often the most active contributors to the project. Once registered as a member of the project team, an individual may add, change, or delete any code in the project repository at will. Third-party support firms may contribute any code that they have developed in the course of their practice-site engagements (for example, to extend the system's functionality), but they are not obligated to submit such contributions to the shared code base.

Commercial Organizations

Commercial organizations are for-profit business entities that directly fund and manage the development of open source software. This model differs from the community organizational model in a number of ways. First, commercial projects are led and directed by an employee of the business entity, typically the CEO or Chief Architect. This person is responsible for directing the product development, growing the user base, and promoting the project. Second, the company employs a dedicated team of developers. These developers work on a full-time basis to add features to the product. Third, product development and feature prioritization are directed through a formal project-management process. As with community-run projects, commercial projects often develop features in response to the requests of specific customers. However, unlike community projects, commercial organizations also have the financial resources to fund the development of features that are specifically requested and sponsored by an individual implementation site. These general purpose features can help to make the FOSS EHR solution a more competitive product, in general. A diagram of a commercial organization support structure is presented in Figure 2. These organizations are similar in structure and operation to firms such as Red Hat Software, the commercial open source provider of Linux.

Figure 2 – Commercial Organization Support Structure



Revenue Generation
Commercial FOSS EHR companies generate revenue primarily through installation and support services provided to physician practice sites. Essentially, the companies give away the software and sell the services. These companies may also generate revenue from licensed resellers who install the software on the company's behalf and remit a share of their revenues to the company in exchange for access to the company's software developers and/or trademarked name. Lastly, many commercial firms also develop and sell add-on modules that enhance the capabilities of the FOSS EHR systems.

Initial funding to develop the open-source software may come from a parent company with experience in supporting other open source projects. Alternatively, a firm that offers support services for an existing FOSS EHR project may adopt and modify that code base and begin offering related support and customization services of its own, often under a different project name. Such forking of projects is made possible by the open source licensing model and has happened a number of times within FOSS EHR projects.

Support and Service Options
As with the community model, the source code for commercial FOSS EHR systems is open and publicly available. Hence, practice sites may contract with independent third-

party support firms for installation, customization, and maintenance services, or they may attempt to implement and maintain the software themselves. Under the commercial model, however, two additional options exist for installation and support services: Practice sites may contract directly with the commercial organization that is maintaining the code or they may contract with a reseller that is officially licensed by the commercial organization.

Many practice sites using commercial FOSS EHR software obtain support directly from the commercial company because this approach has distinct advantages. First, the commercial company is intimately familiar with the ce-base, hardware requirements and configuration options. Second, the commercial company frequently offers proprietary add-on modules that provide useful (even critical) functionality, but are not part of the core source code that is publicly available (GPL licensing allows commercial firms to withhold certain code modules and charge for access to them, even though the modules are technically open source). Third, commercial firms sometimes offer fully hosted versions of their products to customers who do not wish to manage the application and maintain the hardware at their practice site.

Licensed resellers partner with commercial FOSS companies in two ways. First, the resellers are simply sales departments that refer practice sites to the commercial company. In these cases, the commercial company provides the practice site with implementation and support services, and the reseller receives a commission for the referral. Alternatively, the resellers are full-service support organizations that both identify business opportunities and provide all implementation services for the customers. In this case, the reseller pays a fee to the commercial company in exchange for the right to sell their services using the commercial company’s trademarked name.

Source Code Management

With commercial FOSS projects, access to the source code is much more controlled than with community organization projects. Write access to the code repository is limited only to the development team of the commercial organization. Outside developers may contribute to the code base, but they must submit their code through the development team. The development team and the project leader are the gatekeepers of code submission and ultimately decide what code is added to the project.

It is not uncommon with commercial FOSS EHR products for the source code to be partitioned into public and private repositories. Code placed in the public repository is immediately available for the open source community to download and use under the general open source licensing model of the project. The code that constitutes the core architecture and functionality of the system is normally provided in this way.

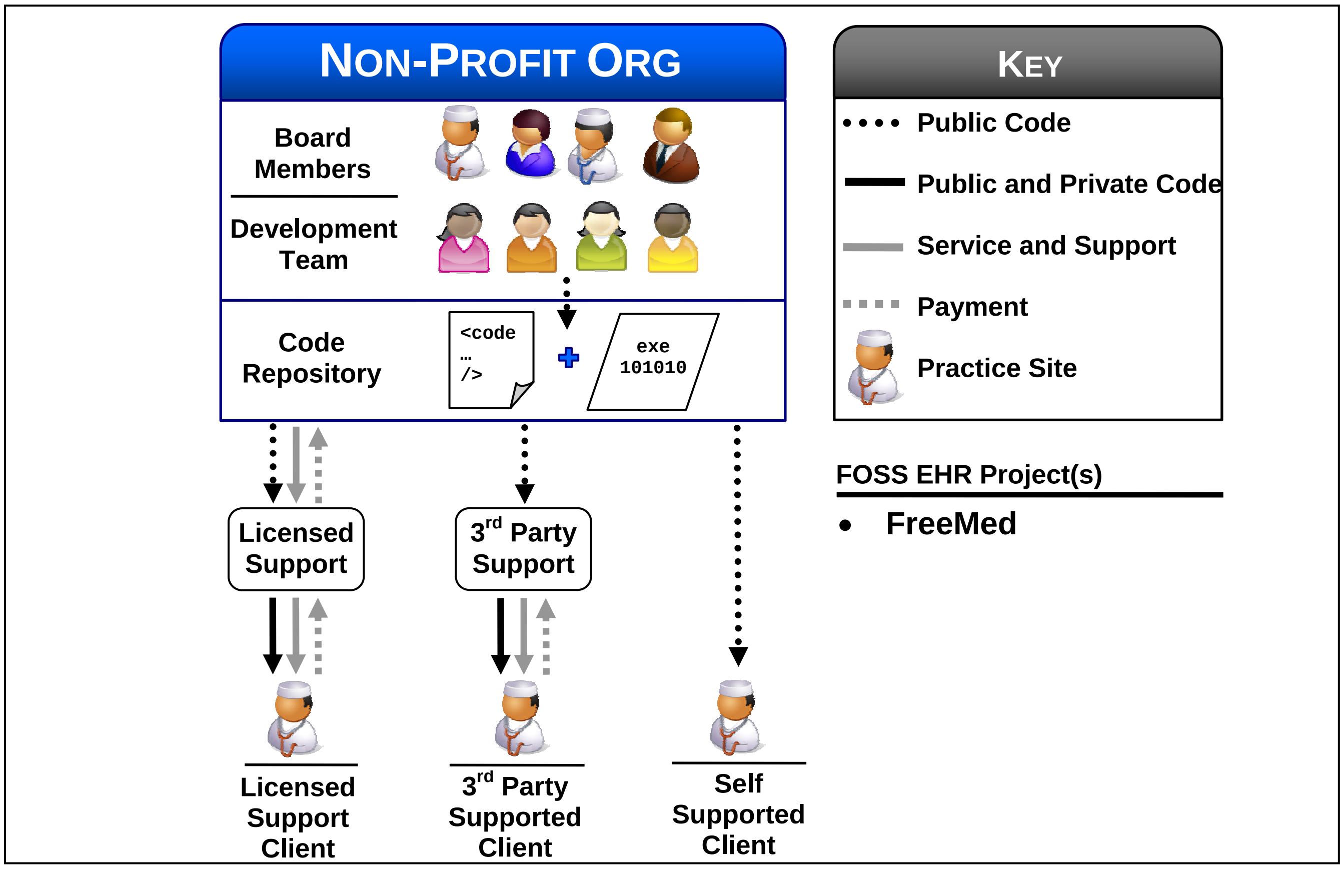
Code placed in the private repository is accessible only to employees of the commercial company. These private repositories are maintained for at least two reasons. First, the development team may be working on a significant change to a core feature of the application and releasing the code to a public repository before it is completed could break the application for all users. In this case, the code will eventually be made public when it is complete. Second, certain code cannot be released to the public repository due to licensing constraints and/or non-disclosure agreements with third parties. In this case, the code is available only to those customers who are willing to pay the commercial company or a third-party whatever additional license fees are required to gain access to the private code. Examples of such code include proprietary decision-support modules for drug-interaction checking.

Nonprofit Organizations

Nonprofit FOSS EHR organizations, like commercial organizations, also have a high level management structure that directs the development of the FOSS EHR application. Projects are often established as foundations and led by a board of directors. Like the senior management of a commercial open source organization, the board prioritizes the development of features and plans the long-term direction of the project. Nonprofit organizations also employ developers on a part-time or full-time basis to perform programming, although the development teams are often smaller. The primary differences between for-profit and nonprofit organizations are the ways in which revenues are generated and new features developed.

A diagram of a non-profit organization support structure is presented in Figure 3. The FreeMed project is an example of a non-profit FOSS EHR project, and its organization is modeled on the Apache Software Foundation.

Figure 3 – Non-Profit Organization Support Structure



Revenue Generation
Nonprofit organizations typically do not to contract directly with practice sites for development or support services. Instead, practices interested in software enhancements make donations to the organization, which it uses to make improvements to the application based on a predetermined product roadmap. The donations are not earmarked for specific tasks, and the improvements thus financed may or may not coincide with the donors immediate needs.

Another source of revenue for nonprofit EHR FOSS projects are fees from licensed third-party support firms that provide installation and support services to physician practices. These firms sign license agreements with the nonprofit organization, which provides them access to the organization's programmers for development consulting and third tier support. In exchange, the licensed firms transfer a percentage of their gross revenues to the project.

Support and Service Options
Practice sites may contract with the nonprofit company directly for installation/support services, if the company offers these services. Otherwise, the licensed support firms are the next closest option. As with the community and commercial models, practice sites may also hire an independent (non-licensed) third-party support firm or attempt to install, configure, and support the system themselves.

Source Code Management
As in the commercial model, write-access to a nonprofit organization's code repository is restricted to the project's development team. Submissions from outside developers, including licensed partners, are reviewed by the development team prior to inclusion in the repository. Nonprofit FOSS projects are more dependent on such outside contributions than commercial FOSS projects, but changes to the code are still closely reviewed and managed by the central organization.

As with all FOSS projects, the project code base is available to the public. Due to the nonprofit nature of the project, the maintenance of a separate public and private repository for code is often unnecessary. For example, the FreeMed Foundation has adopted the policy of producing only code that may be offered under the GPL or LGPL license. This decision may limit the features that the project can offer (because no proprietary modules may be offered as part of the solution), but it guarantees that everyone has equal access to all of the project source code.

Software Architecture, Security, and Remote Access Features
Many FOSS EHR projects are built on the same basic open architecture, known as the LAMP stack. This architecture consists of the following components: Linux for the operating system, Apache (v2) as the web server, MySQL (v5) for the database, and PHP (v5) for the server-side business logic. All five of the projects that this study reviewed in detail are built on the LAMP stack and run on the latest versions of the LAMP components.

All five of the five projects that this study reviewed in detail also use the same access control technology for developing role-based access privileges -- phpGACL (PHP Generic Access Control Lists - http://phpgacl.sourceforge.net/). This technology is also open source software built on the LAMP architecture. When properly configured, phpGACL provides HIPAA-compliant role-based access control to a PHP application's data. phpGACL is a generic permission system that can be used by any PHP-based web application to control access to the application's forms and fields and that can be configured at the account, role, and/or user levels.

The phpGACL application is a completely separate application that is installed on the same server as the EHR. The EHR communicates with the permissions system at the time a user accesses the EHR to determine whether the user has access to the requested page or data. Based on the user's permissions as defined in the phpGACL system, the EHR appropriately filters the pages, links, and data presented to that user. Administration of phpGACL is complex, and it is generally not practical for practice sites to modify permissions themselves. However, the use of this sub-system by FOSS EHRs provides the necessary security to control access to sensitive patient data and to meet HIPAA guidelines for patient data security.

Finally, the web-based architecture of most FOSS EHR systems (including the five reviewed here in detail) enable remote access to the EHR from a provider's home or multiple practice sites. However, the decision to actually enable this feature is made by each practice site, based on its own preferences regarding convenience and security.

Software Deployment Mechanisms

The five FOSS EHR systems reviewed in detail offer various implementation options for practice sites. These options include (1) on-site installation of the EHR software only, (2) on-site installation of computer appliances with pre-installed configurations of the hardware and software needed, and (3) remote web-based hosting of the EHR system, with no on-site installation required. The availability of these options varies across projects and may affect the ease with which physician practices can adopt specific FOSS EHR systems.

On-site Installation

The most common approach to FOSS EHR implementation is to install the application on a server located at the practice site. Under this model, the practice site is free to choose a hardware configuration, operating system, and network infrastructure that work best for the practice (although a third-party support firm may actually purchase and configure the hardware). This option is available because every FOSS EHR system can be installed on a wide variety of operating systems, including Linux, Windows, Unix, and even Macintosh OS X.

Pre-built Appliances

Alternatively, the support vendor may deliver a server with the FOSS EHR application already fully installed and configured to the practice site's specifications. Under this model, the appliance is simply a convenient delivery medium for the software. This model has the added benefit that it ensures that the server hardware specifications and configuration are appropriate to run the application at the desired level of performance.

Remotely Hosted Solutions

Recently, commercial vendors of FOSS EHR project have begun offering completely hosted solutions to practice sites. Under this model, the vendors implement a virtual server at the hosting site and create the necessary user accounts, at which point the practice site is immediately ready to use the system. Vendors that offer this service typically charge the practice a monthly fee (between $50 and $150 per user per month). These vendors manage all aspects of the network security, server administration, and data backup for the remote implementation. The vendor may operate its own data center, or sub-contract management to a hosting service (such as INetU).

Remotely hosted solutions offer both advantages and disadvantages. The chief advantage is that the practice site does not need to manage a server installation or network. Secondly, the start-up costs and effort are typically much lower than for on-site installations. A hosted solution can be a convenient and affordable option for a small practice, especially if a small number of accounts are needed.

At the same time, patient data are not maintained within the walls of the practice, and this can be a significant obstacle for private practice physicians who are concerned about the safety and security of their patient records. To address these reservations, vendors who provide hosted services offer DVD backups of a practice’s data on request and follow standard web-security practices. Second, the customization options are limited when an EHR application is remotely hosted. Vendors typically offer a standard installation with only a few optional modules or forms. Extensive customization to fit a particular need of the practice is not possible. Third, any disruptions in internet connectivity can leave a practice without access to their patient’s data. This risk may be mitigated by maintaining redundant internet access mechanisms (such as cable and DSL), although this approach somewhat increases the total cost of the solution.