Journal of Digital Imaging

A Suggested Classification Guide for PACS Client Applications: The Five Degrees of Thickness

Christopher Toland, Christopher Meenan, Michael Toland, Nabile Safdar, M.D.,
Peter Vandermeer, M.D., and Paul Nagy, Ph.D.

This article defines and describes the numerous types of "clients" for picture archiving and communication systems (PACS). A radiologist uses a client to view images stored in the system. Many PACS are available in the market, and each offers different methods by which a client can view images from the server. The terminology used to describe these different methods can cause confusion and lead to poor choice for those imaging team members who are given the task of purchasing, implementing, and supporting the PACS. We propose a classification of clients with respect to their impact on client work stations, an effect often referred to as the application's thickness. The thinner the client, the less effect it has on the hosting work station. In contrast, a thick client consumes the work station's resources and often prevents a work station from being used to effectively run anything other than the client application. Functionality and supportability are highlighted as key and interacting metrics in determining optimal correct PACS solutions. The importance of a clear understanding of the needs and requirements of all users as well as the client application is emphasized. This relationship between supportability and functionality becomes increasingly important as the industry shifts to enterprise information technology solutions.

KEY WORDS: Enterprise clients, PACS, functionality, supportability, thin client, thick client

INTRODUCTION

What is a client? Long ago, in the days of mainframes and centralized computing, clients were merely dumb terminals with a keyboard attached, designed to send characters back upstream to the mainframe.[1] The logic resided on the mainframe, and the dumb terminal was for presentation only. With the introduction of personal computers (PCs), clients acquired limited processing power. With the ability to wrap the data stream coming from and going to the mainframe, clients could perform some actions with data either before or after presentation. Mainframes, however, still maintained the processing power, and clients remained in the early stages of development.

A signal leap in technology came with networked clients. Freed from serial connection, users could receive the data stream over the network from the mainframe. As the processing power of clients increased, more mainframe processing and logic were pushed out to the client. Decentralization resulted, as mainframes were no longer needed and clients became silos of processing—an arrangement that came to be called "distributed computing". As a computing society, we are moving to a more distributed computing model. This trend in decentralization allows multiple servers to perform different tasks within a single solution rather than relying on one mainframe process. The central computing question, whether

This study was not supported by grants or other outside assistance.

From the Department of Diagnostic Radiology, University of Maryland School of Medicine, 22 S. Greene St., Baltimore, MD 21201, USA.

Correspondence to: Christopher Toland, Department of Diagnostic Radiology, University of Maryland School of Medicine, 22 S. Greene St., Baltimore, MD 21201, USA; tel: +1-410-3281054; fax: +1-410-3280641; e-mail: ctoland@umm.edu

Online publication 04 October 2006
doi: 10.1007/s10278-006-0930-z

in radiology or other fields, becomes: Where are the data? Either the data are on the server, ready to be manipulated there, or they must be retrieved to the client for manipulation. This move to a client/server architecture has led to a crucial juncture in client development, with special implications for medical imaging.

The holy grail of enterprise client development has been the development and implementation of a lightweight "thin client" with maximum functionality. In the world of radiology, the realization of this elusive goal would require the power of a diagnostic work station with advanced image processing and manipulation tools that could run inside a standard browser on any operating system. Over the past 10 years, various approaches to this goal have been proposed, each with idiosyncratic (and often confusing) terminology and distinct advantages and disadvantages. We describe the five major types of clients commonly in use today with picture archiving and communication systems (PACS) and discuss their suitability for different types of users. The goal is to provide information that can assist decision makers in imaging departments in evaluating new PACS systems. In so doing, we describe a novel system of classification of common architecture for these systems and discuss specific terms that may prove confusing or misleading in analyzing their suitability.

DISCUSSION

The tradeoff facing the design of PACS clients today is between the functionality of the client and the ability to deploy and support this functionality throughout a heterogeneous enterprise that is difficult to keep under tight systems control. The system for classification of common architectures proposed here is based on this tradeoff (Fig. 1). The general rules of this architecture are consistent, but individual implementations will vary

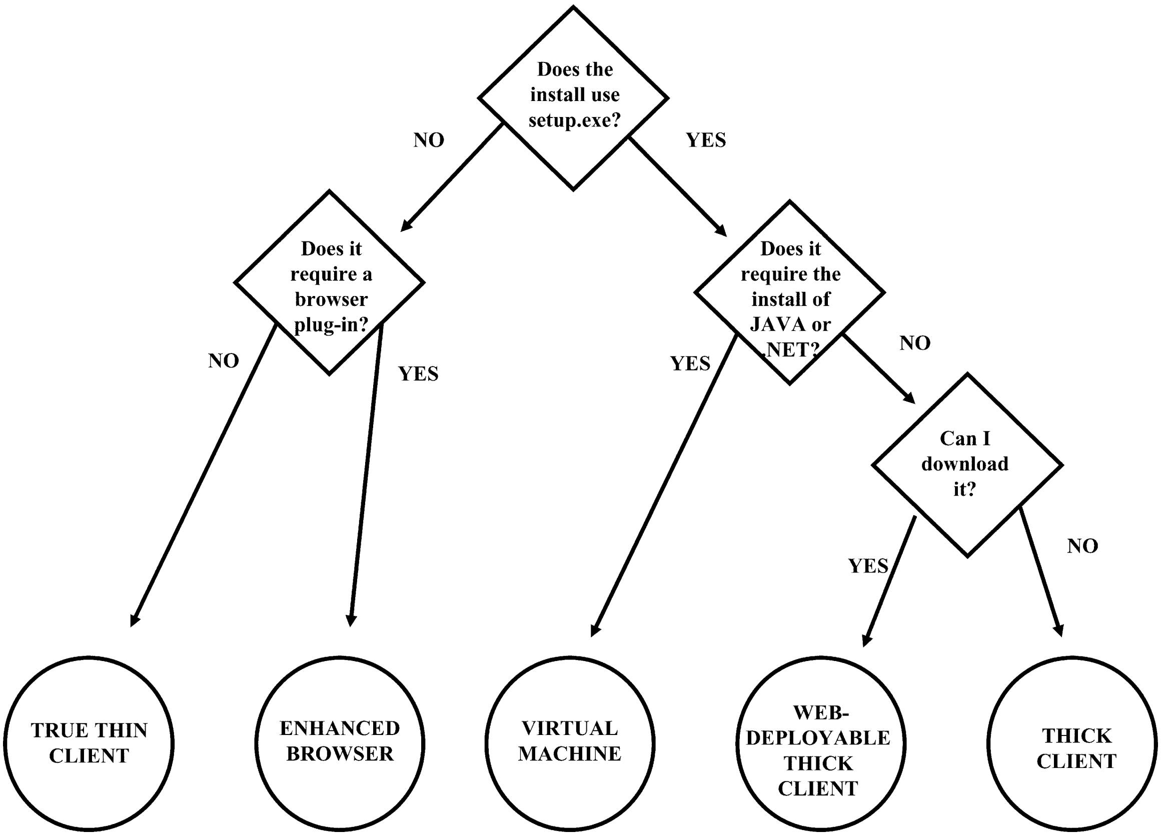
Does the install use setup.exe?
NO
YES
Does it require a browser plug-in?
NO
YES
Does it require the install of JAVA or .NET?
YES
NO
Can I download it?
YES
NO
TRUE THIN CLIENT
ENHANCED BROWSER
VIRTUAL MACHINE
WEB-DEPLOYABLE THICK CLIENT
THICK CLIENT

Fig 1. What is your client? This decision tree of questions to ask vendors is designed to help determine the "thickness" required.

depending on the ways in which vendors choose to deploy specific technologies.

Different types of users of an enterprise PACS have significantly different requirements for levels of functionality and supportability. No single technology stack is perfect for every user in a typical enterprise PACS. Instead, tradeoffs must be made between supportability and functionality as the two major criteria against which various architectures are judged (Fig. 2). Conceptualizing the interplay of these two criteria also helps to illuminate the problems that are introduced with current solutions to these tradeoffs. The five degrees of thickness currently represented in these client solutions are as follows: thin clients, enhanced browsers, virtual machines and smart clients, web-deployable thick clients, and thick clients.

Thin Clients

"Thin client" is among the most misused and overloaded terms in computing. True thin clients utilize standard browsers and standard plug-ins. Differences in client platform (PC vs. Mac) should have no effect on functionality and should not require different clients. These clients should leave no files or changes on the host computer. The advantages of thin clients are lower hardware requirements, with most of the processing moved onto the servers.[2]

An example of a thin client is a Web-based e-mail client, such as gmail, that is completely browser- and client hardware-independent. Despite widespread misconceptions to the contrary, no current PACS clients are true thin clients, because of the need for image visualization and manipulation at the work station.

Enhanced Browser

Enhanced browsers require the installation of ActiveX controls or Java applets. ActiveX controls can run only in a Microsoft Windows environment, whereas Java applets can be platform-independent. PACS vendors can create ActiveX controls and Java applets for execution by the browser. An example of an ActiveX control is Windows update, which has the capability to scan the host and install needed updates and patches. PACS clients based on ActiveX have more functionality than thin-client applications but less than thick-client applications.

Virtual Machines and Smart Clients

Virtual machines function at a layer above operating systems to provide a consistent application environment, allowing vendors to create applications without concerns about operating system stability (Fig. 3). The most dominant virtual machines are Java by Sun Microsystems and .Net by Microsoft. Microsoft's .Net Framework, unlike Java, only runs in the Microsoft Windows environment. This software contains many built-in functions and provides a buffer between the developer and the underlying operating system. Virtual machines remove the need to create a client for each platform, putting the focus on a single instance of a client instead of multiple instances of the same client in multiple programming languages.

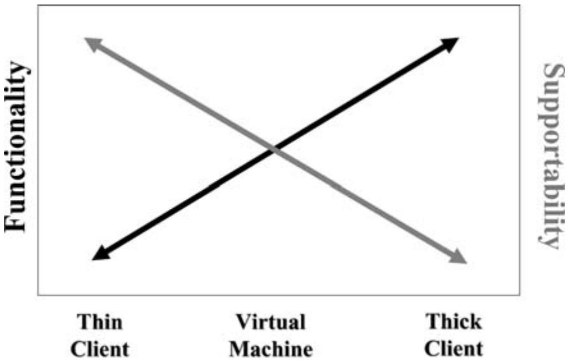

Fig 2. Relationship of functionality and supportability with respect to thickness. The thinner the client, the lower will be its functionality—but it will be easier to support. The thicker client, although harder to support, has more functionality.

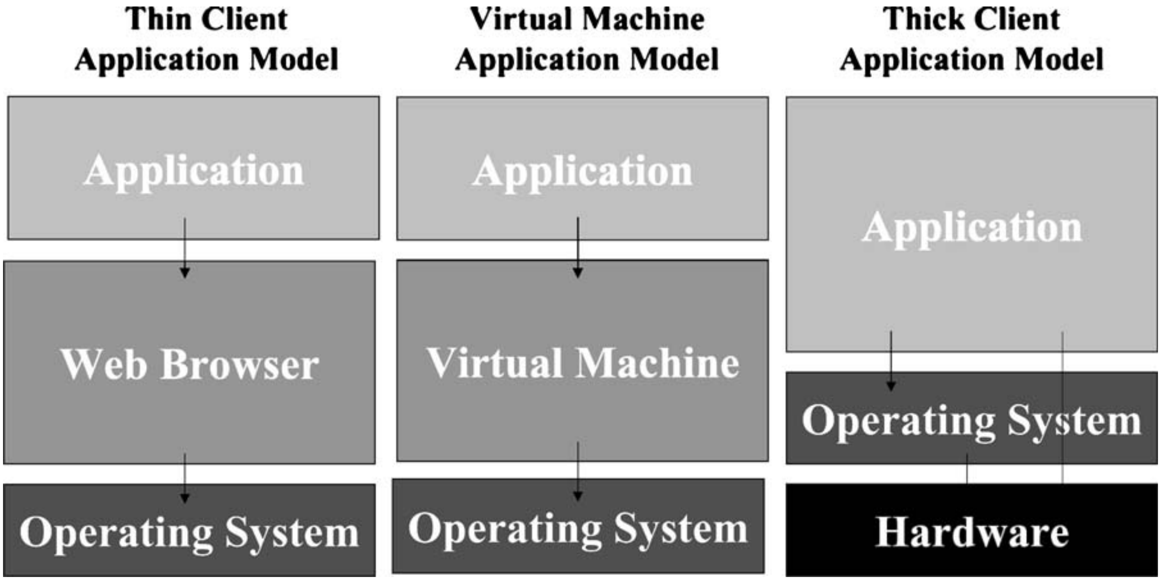

Fig 3. Application models of a thin client, a virtual machine such as Java or .Net, and a thick client. These models display the interaction between the different layers of an application and the operating system.

Web-Deployable Thick Clients

These are downloadable, self-contained installation files containing both executables and dynamic link libraries (DLL), typically delivered in a setup.exe configuration. DLL libraries are the focus of frequent conflicts between applications sharing the same DLL. As more than one industry analyst has noted, as the complexity of applications and client platforms has increased, the difficulties associated with deploying applications to client machines in a reliable and secure way have become more challenging. One example was that of "DLL Hell", where one application could "break" another by deploying an incompatible shared component or library.[3] These clients would be acquired and upgraded through the Web. These clients should be aware of their current version and prompt the user when new upgrades become available. The client should be small enough for Web downloading to be feasible. An example of a Web-deployable client would be Mozilla's Thunderbird mail client. In comparison with the Web-based thin client, this type of client has considerable additional functionality. Because it is small in size and easily upgradable, this type of client is often mislabeled as a thin client. However, these applications still install the same types of files, executables, and libraries as the final type of client to be discussed here, the thick client.

Thick Clients

Thick clients have specific hardware requirements for providing advanced functionality. These may include requiring a specific hardware platform, video cards, or others. Thick clients are usually massive in size, but size does not indicate degree of thickness. These types of clients are most commonly installed via compact disk disks, and installed files would include executables, libraries, and configuration files. Thick clients sometimes require advanced configuration, often by the vendor. An example of a thick client would be Microsoft's Outlook. Outlook requires a compact disk to install, has too many registry entries to count, and installs hundreds of files.

Supportability is a major decision factor when selecting a category of PACS clients. Thin clients require minimal end-user support at the desktop and offer a much easier approach to support new operating platforms, because maintenance can occur at the server level rather than requiring a physical visit to each desktop.[4] Enhanced browsers extend to plug-ins in the browser that may interfere with the operating system. Windows patching and other plug-ins may complicate supportability. Virtual machines add yet another layer of complexity, requiring increased troubleshooting time to resolve more complex issues, such as version conflicts. The vendor might have issues surrounding version control, requiring additional time for validation from the vendors as updates to Java and .Net are released. Web-deployable thick clients and traditional thick clients have comparable serviceability, because they each require standard vendor application support. Thick clients may also have some direct hardware dependencies and have the highest support costs.

From a user's perspective, functionality is the most crucial determining factor for a PACS client. For decision makers, functionality and supportability are often located at opposite sides of the spectrum. Thin clients are limited in functionality because of a lack of resources from the operating system but are easy to support. Enhanced browsers through plug-ins, such as activeX controls, add some of the functionality that thin clients are missing. Virtual machines, .Net frameworks, and enhanced browsers are all capable of appearing to have the same functionality to the end user. Thick clients provide the most functionality by building directly on top of the operating system and, in some cases, creating their own device drivers. Vendors usually tie their thick clients to a specific hardware platform to provide additional functionality and increased performance. Performance of a PACS client has a direct effect on user acceptance.

No single solution fits all users perfectly. It is important to match the right client with the right users. Figure 4 illustrates potential problem areas for scrutiny during technology assessment. Users can be assigned to three basic groups when assessing PACS client needs: radiologists, power users, and enterprise users. For the radiologists' work station, a thicker client will often suffice, because only a few dozen work stations must be supported. Having a Web-deployable work station is more desirable, because this lowers support needs compared with those of a true thick client and also facilitates teleradiology. Power users are those individuals throughout the enterprise who

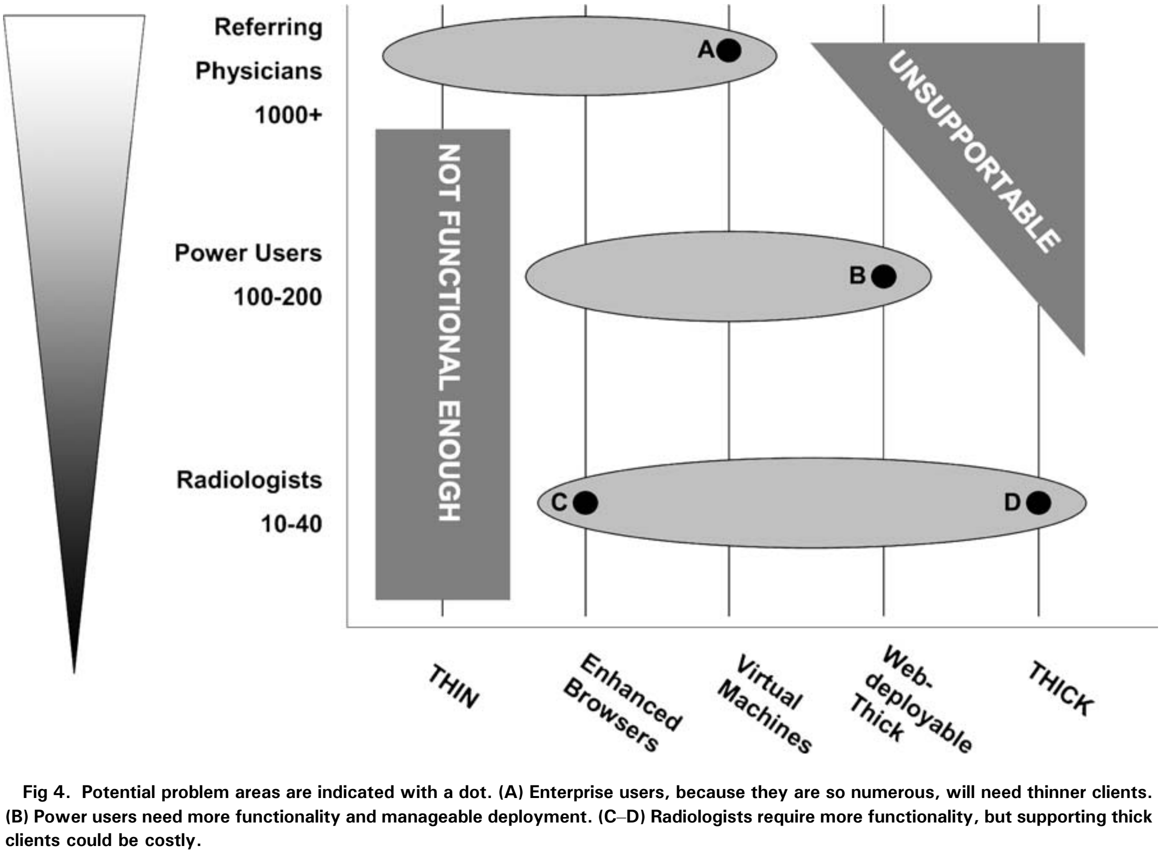

Fig 4. Potential problem areas are indicated with a dot. (A) Enterprise users, because they are so numerous, will need thinner clients. (B) Power users need more functionality and manageable deployment. (C–D) Radiologists require more functionality, but supporting thick clients could be costly.

want to have the same functionality and usability as the radiologist. They typically represent about 10% of the enterprise user base and incorporate high use areas, such as surgery and the emergency department. PCs in these locations may be under system management control by a hospital information technology group, which can lower the cost of maintenance. Enterprise users encompass the rest of the individuals who will want occasional quick access to reports and key images. Many PCs in the enterprise are likely to be under no real consistent system management control, making thicker clients quite difficult to support for the majority of users.

Figure 4 also indicates potential problem areas (red dots). For enterprise users, which can number into the thousands in some institutions or systems, deploying virtual machine software can increase front-end support cost. Maintenance of those user's applications, such as handling version conflicts, will elevate the ongoing support cost. The blame for such conflicts may come to rest with the support organization. For power users, Web-deployable thick clients can be difficult to deploy and even harder to continuously manage. Maintenance and upgrades can weigh heavily on annual support costs. The radiologist user groups faces two problems. First, enhanced browsers may not have sufficient power and functionality for the radiologists' tasks. Second, hardware and vendor maintenance associated with thick clients make them prohibitively costly. Moreover, thick client applications do not facilitate teleradiology.

CONCLUSION

A thorough understanding of clients is crucial in choosing an appropriate and effective PACS solution. A multipronged approach promises to provide solutions that directly address the diverse needs of the various users in the enterprise. Understanding what questions to ask vendors, how to weigh trade-offs between functionality and supportability, and the real-world differences between types of clients will help to more easily identify problem areas and plan for implementation of new technologies. These questions should be asked as early as possible in the process of identifying and acquiring new PACS. Moreover, finding answers to these questions should be a group effort: no users or support groups (whether at the department or enterprise level) should be blindsided by new and complex rollouts. It is imperative to get people involved and not to blindside your enterprise IT department with a massive rollout that they will have to validate. Finally, the effort to better understand the role of clients in PACS requires active engagement of and communication with vendors, who should be urged to create clients that respond to and address specific problems without the installation of additional independent systems with their respective data silos.

REFERENCES

1. Schmelzer R: You can never be too rich or too thin. searchWebservices.com. March 22, 2004. Available at http://searchwebservices.techtarget.com/tip/1,289483,sid26_gci956073,00.html?FromTaxonomy=%2Fpr%2F292241. Accessed on 22 May 2006

2. Buxbaum PA: Thin-client diet. *Military Information Technology.* May 2, 2006. Available at www.military-information-technology.com/article.cfm?DocID=1422. Accessed on 22 May 2006

3. Hill D: What is a smart client anyway? David Hill's Weblog. February 2, 2004. Available at http://blogs.msdn.com/dphill/articles/66300.aspx. Accessed on 22 May 2006

4. Bertolini P: Thinking thin. *CIO.* January 15, 2005. Available at http://www.cio.com/archive/011505/peer.html. Accessed on 22 May 2006