Developing the Perfect PACS

Using LEADTOOLS PACS Imaging for Building Robust PACS Solutions

Introduction

Since 1997, LEAD Technologies has been creating tools for the Medical Imaging industry and has compiled its expertise and customer feedback into a diverse, robust, and powerful PACS Imaging SDK which you can use to develop a wide array of PACS applications. Due to the vast number of features and interfaces, it can be difficult to know where to begin. This document is designed to help you decide which is best for you.

LEADTOOLS PACS Imaging contains four interfaces for PACS development: DICOM Communication Engine, PACS Framework, Workstation Framework, and Medical Web Viewer Framework. Each engine and framework provides unique features and depending on your objective, your DICOM experience and the amount of customization you want to do, one engine may be a better option than another.

DICOM Communication Engine

The DICOM Communication Engine provides a development framework for DICOM Message Exchange. Through the DicomNet class, it includes a programming interface for Association, DIMSE-C, DIMSE-N and the underlying TCP/IP communication. The DicomNet class includes support for asynchronous or synchronous TCP/IP for multiple clients and servers. To keep your transmitted data private and secure, LEADTOOLS DICOM Communication Engine classes utilize TLS and ISCL security profiles. It is designed with the ever-evolving DICOM specifications in mind, giving you low level functions and flexibility to implement new DICOM services as they come. DICOM Communication Engine also provides a high level DicomPrintScu class to ease the development requirements of working with a DICOM Print SCP.

The best part about DICOM Communication Engine is that you have full control over all features and can make an application that is catered to your exact needs. For example, an advantage here is that you can get the highest level of logging details since there are more events to which you can subscribe.

PACS Framework

The PACS Framework builds upon the DICOM Communication Engine classes by implementing high level client and server classes. For the client, PACS Framework provides classes to do common command messages such as C-STORE, C-FIND, and C-MOVE. Instead of requiring you to write code for the connection, association, and message handling processes, all of this is done internally. For example, the StoreScu class has a Store function where you pass the server's information and the name of a DICOM file and the StoreScu class will handle all of the low level communication for you.

The PACS Framework server features are designed to alleviate association and connection management, manage multiple servers, and organize enterprise workflow with less code and more scalability. To increase performance, organization, and ease of maintenance, you can host multiple SCPs on the same machine and each one will run as its own Windows Service. A Service Manager application and source code is included which supplies a simple user-interface for adding, modifying, and maintaining the DICOM servers. These servers will take care of the low level communication, threading, and concurrent client connections so all you have to write are add-in libraries to handle events for related DIMSE service commands like C-FIND, C-MOVE, and C-STORE. You can also break down the addins by SOP class and Transfer Syntax. A useful situation for this would be if you wanted to use separate databases and workflow for different departments such as Cardiology and Oncology. The LEADTOOLS PACS Framework includes fully functional sample add-ins for logging, secure communication, Study Root Query/Retrieve, Storage, Storage Commit, Modality Work-list (MWL), Modality Performed Procedure Steps (MPPS) services and a rule based DICOM Router.

The PACS Framework requires much less knowledge and experience with DICOM messaging and Windows Service application development. Therefore you can devote more time to improving your user interface and database rather than learning and working with DICOM. Since the PACS Framework client classes are built upon the DicomNet class, you can still gain access to lower level features to provide you with more power and flexibility.

PACS Workstation Framework

The PACS Workstation Framework is a viewing solution as opposed to a PACS solution. It is comprised of a set of .NET UserControls and components for easily developing a full-scale Medical Workstation application. In fact, the workstation demo that ships with the PACS Imaging SDK is an end-user application that can be skinned and shipped. PACS Workstation Framework is a combination of a radiologist style viewer, PACS Client and PACS Server applications which are based on the PACS Framework client and server tools. The viewer comes with common tools for measurement, window level, stack, cine, scale, magnify glass, ROI, annotations, reference lines, MIP, MPR and 3-D volume reconstruction. PACS Workstation Framework can query any PACS server to retrieve and display images. Images retrieved from another PACS will be stored to its own local database for quick re-loading. Its local database is also a PACS server from which which other workstations throughout the hospital or office can query and retrieve its images.

The overall purpose and advantage with the Workstation Framework is that you already have an end-user application built for you. Therefore little to no DICOM experience and coding are required. The PACS Workstation Framework is designed as separate customizable components which can be used together or independently. A demo that combines all the components together to form an end user application is provided with source code, so it can be modified to suit your specific needs.

Medical Web Viewer Framework

The Medical Web Viewer Framework is similar to the Workstation Framework in that it can be used as a ready-to-ship end-user application. Medical Web Viewer Framework is a fully featured and secure ASP.NET web application that uses WCF (Windows Communication Foundation) services on the server to communicate with and stream images to a .NET Rich Client viewing application over the internet or intranet. The included WCF services provide Query, Retrieve, Store, and Manage services. The viewer comes with the same set of popular radiologist tools as the Workstation Framework’s viewer. The WCF image streaming allows the customer to download large series while still using the viewer. To reduce memory usage and transmission speed, low resolution thumbnails are downloaded first with full resolution images loaded on demand. Downloaded images are securely cached on the client's machine, so they can be loaded faster the next time they are viewed.

An example database is provided so you do not need to make any changes and you can deploy the application as-is. However, the web application's source code is provided if you desire to make customizations or write new plugins. The Medical Web Viewer Framework solution is separated into several projects and layers. An advantage to this is if you already have an existing database or PACS, you could just change the Data Access Layer (DAL) used by the WCF services. This would make the switch transparent to all other aspects of the application.

Conclusion

In the software development world, there are often multiple ways to accomplish the same goal. LEADTOOLS PACS Imaging provides different avenues to cater to your style, needs, and development time frame. The Dicom Communication Engine gives you all the features and detail you need to create a robust application to your exact specifications, the PACS Framework encapsulates common features into easy to use classes that save development time and resources, and the Medical Workstation and Medical Web Viewer Frameworks provide ready-to-ship viewing solutions that can be modified and extended. If you have more questions about which technology is best for your application, speak with a LEADTOOLS Developer Support representative. LEADTOOLS PACS Imaging is the stepping stone to development success you need to provide a PACS solution with more flexibility, dependability, and features in exponentially less time.