WHITE PAPER

JPEG2000
for Medical Imaging

AWARE

05.13

The healthcare industry requires modern open standards-based image compression techniques optimized for medical data and workflows implemented by complex medical systems that acquire and manage digital medical imagery. These requirements led to the adoption of JPEG 2000 as a wavelet compression standard (ISO/IEC 15444-1) and its inclusion as a valid compression method in DICOM.

Reference implementations of the JPEG 2000 ISO standard, or baseline implementations, are designed to address a sampled cross section of encoding and decoding options to provide a reference of how the standard could be used. However, these reference implementations are not optimized and their feature sets are not flexible enough for specialized medical applications.

Aware, Inc. was the first commercial organization to provide wavelet based compression algorithms as commercial off-the-shelf libraries designed for integration in larger systems. Twenty years of R&D combined with a strong commercial focus have kept Aware in the forefront of the wavelet based image and data compression market. Today, wavelet compression techniques are ubiquitous across a spectrum of medical imaging systems that require compression and streaming.

This paper talks about some of the major features of the Aware JPEG 2000 implementation:

- **Direct support for DICOM objects**
- **Compress to quality target**
- **Fully-compliant fixed point decompression**

The paper also introduces features of the JPEG 2000 standard such as 3D volumetric compression and JPIP that are applicable to medical applications.

Aware's JPEG 2000 Implementation

The Aware initial implementation of the ISO/IEC 15444-1 JPEG 2000 standard was designed and implemented by a team of specialists with more than a decade of experience providing compression/decompression, image viewing and image processing software for medical OEMs. Incorporating almost twenty years of experience, the API brings unprecedented reliability and expertise to users. The API is clean, easy to understand, and easy to implement. Both the encoder and decoder are optimized through a combination of algorithmic and coding techniques that efficiently utilize the design of modern CPUs.

DIRECT SUPPORT FOR DICOM

The Aware JPEG 2000 API fully supports the input and output of single and multi-frame DICOM formatted data. The API will parse, encode, decode and reformat the DICOM data. JPEG 2000 support was added to the DICOM standard in November 2001 with the publication of "Digital Imaging and Communications in Medicine (DICOM) Supplement 61: JPEG 2000 Transfer Syntaxes." As described in Supplement 61, WG 4's motivation was based on the fact that "…there are both real and perceived limitations" with traditional JPEG. "…WG 4 began to investigate alternatives, particularly those based on wavelet transformation, multi-resolution analysis and more sophisticated entropy coders than Huffman coding." The adoption of JPEG 2000 as an ISO standard provided DICOM with a timely solution. Both the JPEG 2000 lossless compression and lossy compression algorithms have been accepted for use.

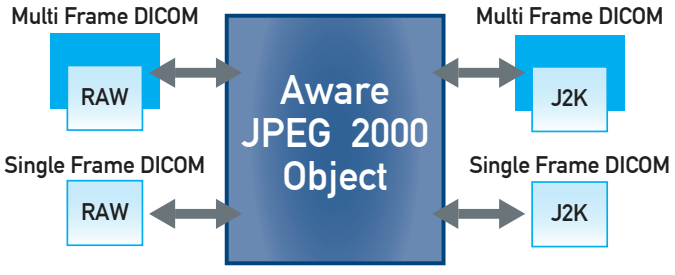

Figure 1 - The Aware JPEG2000 library includes native support for DICOM input/output

Aware supports the following transfer syntaxes, enables transcoding between them and can extract image data as separate files:

- **DICOM with embedded raw image**
- **DICOM with embedded JPEG 2000**
- **DICOM with embedded JPEG or Lossless JPEG**

Aware's JPEG 2000 library enables full DICOM file parsing, access to the most commonly used tags and a data dictionary that supports the most commonly used tags. The Aware JPEG 2000 software also provides support for multi-frame DICOM files and full support for all of the image color spaces used within DICOM.

COMPRESS TO TARGET QUALITY

The process of selecting a target compression ratio (e.g. 10:1) or target file size (e.g. 64 kbytes) does not always correspond to an accurate measure of image quality. An acceptable compression ratio of X for one image type (e.g. CT) may introduce an unacceptable degree of image degradation for another image type (e.g. MR). A useful feature in the Aware JPEG 2000 API is the ability to encode an image to a specified target image quality.

The JPEG 2000 standard includes lossy and lossless compression techniques. Lossy compression results in smaller data size but, depending on the amount of compression, may result in image degradation. Lossless compression will maintain 100% image integrity but is limited to compression ratios of 2:1 to 3:1 on average.

The use of lossy compression techniques with medical images requires controls to maintain image quality. A commonly used practice is to select a target compression ratio or a target file size that has been predetermined to provide diagnostic quality. The optimal compression ratio is chosen through empirical methods and is influenced by the characteristics of an image. The size, resolution, dynamic range, bit depth, noise content and frequency components of an image affect its compressibility.

This process of optimal compression ratio selection for each modality type can be further modified to reflect how the images will be used (fast browsing, reference viewing or diagnostic reading), who the user is (radiologist, clinician, insurance provider) and the characterization of system resources such as available bandwidth, memory, and processor.

Research attempts to closely correlate expert human measures of image quality with that of a machine. The field of image processing and compression typically uses methods that compare each pixel of the original, uncompressed, image with the same pixel of the compressed image. Images with large differences across large numbers of pixels are considered to have been degraded more than an image with small differences across fewer numbers of pixels.

A common quantitative measure that uses this technique is called the "mean square criterion". In many image-coding applications, the mean square error is often expressed in terms of a peak signal-to-noise ratio, pSNR. The Aware API functions implement this pSNR measurement.

Aware's JPEG 2000 API includes functions that enable an image to be compressed to a target pSNR value which is a machine measurable quantity of image degradation. Both the entire image can be compressed to a target pSNR and each quality layer can be coded to its own pSNR value. The ability to encode each quality layer to a separate quality level can be used with the progressive display capabilities of JPEG 2000 to set minimum image quality values for each layer that is transmitted and displayed.

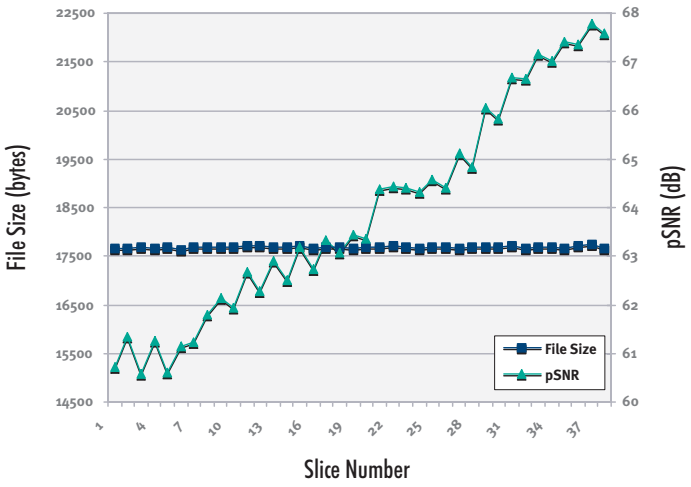

Figure 2 - Compression to Target File Size: Constant compression results in different levels of image distortion for each image in a CT series.

Figure 2 shows a graph of file size vs. pSNR for a series of 38 CT images from one study. All of the images were compressed using Aware JPEG 2000 and a target compression ratio of 30:1. The original images have a dimension of 512 x 512 pixels and a bit depth of 16 bits per pixel. The data shows that while each image was compressed to the same target file size, the resulting pSNR, or distortion varies from image to image.

Figure 3 shows the same series of images compressed to a target pSNR of 64 dB. The resulting compressed images were within a 0.3 dB range of the specified target pSNR while the resulting file sizes varied +20% to -20% around the mean file size. Setting the image distortion level to a constant pSNR results in different levels of compression for each image in a CT series.

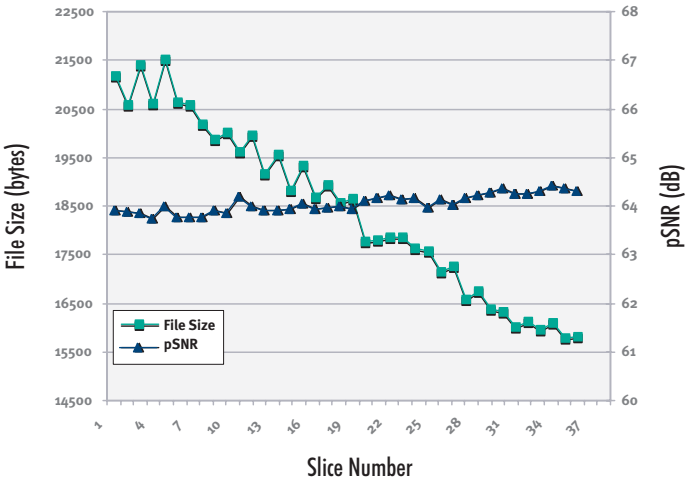

Figure 3 - Compression to Target Quality:

The Inverse of Figure 2. A constant image distortion level set to a constant pSNR results in different levels of compression for each image in a CT series.

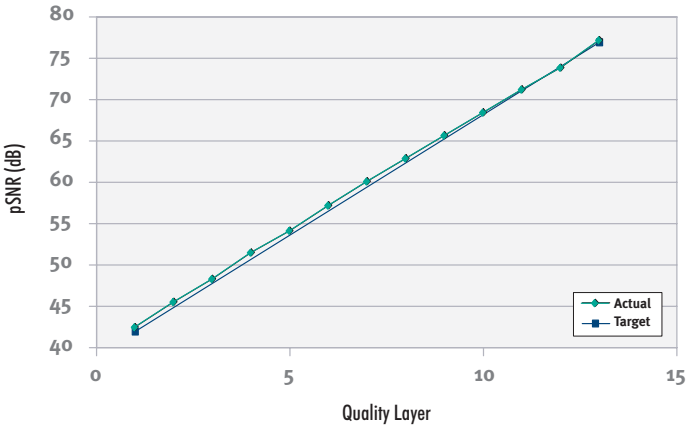

Figure 4: Target pSNR by Quality Layer:

The Aware J2K encoder enables very accurate pSNR distortion control for each quality layer of an Image. This graph shows specified and actual pSNR values for each layer.

Aware's technology also provides a method by which to specify a target pSNR for each quality layer within the JPEG 2000 file. Figure 4 shows a plot of the actual pSNR and the Target pSNR for a JPEG 2000 file with 10 quality layers.

The Aware J2K Encoder enables very accurate pSNR distortion control for each quality layer of an image. This Graph Shows Specified and Actual pSNR Values for each layer. Baseline or reference implementations of JPEG 2000 may use one of two methods to specify the amount of compression applied to an image, compression ratio, or target file size.

The Aware implementation provides an additional method: compression to target pSNR. The Aware "compress-to-target pSNR" functionality provides a robust and repeatable method to compress medical image data with a machine measurable quality metric.

FULLY COMPLIANT FIXED-POINT DECOMPRESSION FOR OPTIMUM PRECISION AND SPEED

Medical imaging systems and JPEG 2000 require precise decoding of image data such that an ISO-compliant JPEG 2000 decoder passes precision testing. To do this, developers need to design an optimized system by balancing precision and the speed of data through a processor's pipeline. To understand how, it's important to look at fixed point and floating point arithmetic.

Most modern processors and compilers support both fixed point and floating point arithmetic. Floating point arithmetic performs calculations with greater precision, typically at the expense of data throughput because floating point numbers must be expressed with extra bytes. The compactness of fixed point arithmetic makes it ideal for resource constrained, embedded applications, or computationally intensive applications such as JPEG 2000 image compression.

Fixed point mathematics is potentially useful for systems which utilize JPEG 2000 because it requires fewer bytes of data per pixel to be processed. The advantage is that more fixed point data can be passed through a processors pipeline. However, the potential downside of fixed point processing is the diminished precision of the data.

The challenge is to design a decoder that meets both criteria of fast fixed point decode and ISO compliance, which is not a trivial endeavor. Reference implementations of JPEG 2000 do not include fixed point processing. Some JPEG 2000 implementations operate in a default mode which is fixed point, but is non-compliant. In order to be compliant, these implementations must be run in something called "precise mode" or "floating point mode." Floating point mode requires more memory and is typically slower on SIMD processors such as Intel's x86 devices.

The Aware JPEG 2000 technology supports both fixed point and floating point paths through the encoder and decoder. The Aware fixed point mode meets the precision criteria for ISO compliance. System designers can use the Aware floating point encoder in the subsystem that compresses the image data. This guarantees that a maximum level of precision is maintained. Viewing workstations can incorporate the Aware compliant fixed point decoder to gain the click-to-view speed advantages available through this method. This fixed point decoder is fast and fully ISO compliant. The advantage to medical imaging systems designers is a full level of compliant encode and decode flexibility to achieve the optimum balance of precision and speed.

Compression of Three-dimensional Medical Image Data Using JPEG 2000 Part 2

The following illustrates the performance of three-dimensional data compression using JPEG 2000 Part 2 and Part 1. For lossless compression, applying Part 2 to take advantage of inter-image correlation achieved a compression ratio that was 15-18% higher than if images were compressed independently. For lossy compression, the results were even more dramatic. At a specific compression ratio, using Part 2 increased the average pSNR of the image sequence by 5-18 dB over the Part 1 case. This translates to an increase in compression of a factor of 2-3 for a given average pSNR value.

For three-dimensional data sets there are Part 2 extensions that allow for the use of several different types of decorrelating transformations in the third dimension, referred to as multiple component transformations (MCT). Specifically, wavelet transforms, linear transforms, and dependency transforms (such as DPCM techniques) are all specified under Part 2.

These multi-component transformations in Part 2 of JPEG 2000 can be effective in compressing volumetric datasets because the correlation between adjacent images can be exploited to achieve better compression than if each image were compressed independently. For lossless compression, the reversible 5-3 wavelet transform is applied in the third dimension. For lossy compression, the non-reversible, floating point 9-7 wavelet transform is applied.

The following addresses the compression of three-dimensional, volumetric image data using Part 2 of the JPEG 2000 standard. It reports on the results of using the wavelet-based multi-component transforms defined in Part 2 to compress volumetric medical datasets. Test results for both lossless and lossy compression are presented.

For lossless compression of three-dimensional datasets, the reversible 5-3 wavelet transform is first applied independently to each pixel in the third dimension. The 5-3 transform is an integer transform, so the data remains in integer format throughout this process. After applying this transform to each pixel, each image in the transformed sequence is compressed losslessly using Part 1 of JPEG 2000. The results are compared to those achieved by applying Part 1 lossless compression to each image independently. Results are summarized in Section 5.1.

For lossy compression of three-dimensional datasets, the irreversible 9-7 wavelet transform is first applied independently to each pixel in the third dimension. The 9-7 transform is a floating point transform, so the data is converted to floating point during this operation. After applying this transform to each pixel, each image of the series is compressed independently using the lossy option of Part 1 of JPEG 2000. The compression results are compared to those achieved by applying Part 1 lossy compression to each image independently. Results are summarized in Section 5.2.

The multi-component transformations in Part 2 also allow for the grouping of components into independent collections (tiling in the third dimension). For each pixel, the multi-component transform is applied to the pixel samples in each component collection independently. The size of the component collection and the specific components that go into each collection are indicated in the header of the compressed file.

Using component collections may be advantageous for 3D imagery that is highly correlated over a small number of components, but less correlated over the entire sequence. Smaller component collections can also be used to limit the amount of memory needed to perform a complete 3D compression (or decompression) on a large data set.

Finally, for the decoder, component collections can be used for improved access to the compressed imagery, as only a single collection must be decoded to decompress a single image in the sequence.

Results of lossless and lossy compression are illustrated below, using component collections of 20, 40, 80 and N frames, where N is the total number of components in the given sequence. The 3D compression techniques described above were tested on three different sets of volumetric medical data. Characteristics of the image sequences are given in the table below:

Table 1. Test sequence information.

Sequence name	Image size	# of images (N)	Bitdepth	Uncompressed sequence size
Seq. #1	256 x 256	127	8	7.9 Mbytes
Seq. #2	512 x 512	449	16	224 Mbytes
Seq. #3	512 x 512	620	16	310 Mbytes

LOSSLESS COMPRESSION RESULTS

Each of the three sequences was compressed losslessly using Part 1 JPEG 2000 and Part 2 JPEG 2000 using components collections of 20, 40, 80 and N, where N is the total number of images in the sequence. The resulting compressed image sizes are shown in the table below.

Table 2. Lossless compression results.

Note that Part 2 JPEG2000 compression reduces the compressed sequence size by 15-18%.

Sequence name	Uncompressed sequence size (Mbytes)	2D Part 1 JPEG2000 compressed size (Mbytes)	3D Part 2 JPEG2000 compressed size, at different component collection sizes (Mbytes)				Improvement with 3D compression, collection size N
			20	40	80	N	%
Seq. #1	7.9	3.81	3.27	3.25	3.24	3.23	15.2
Seq. #2	224	75.8	62.7	62.3	62.1	61.8	18.5
Seq. #3	310	120	105	104	101	100	16.7

Note that JPEG 2000 Part 2 compression reduces the compressed sequence size by 15-18%.

Reviewing the lossless results presented in the Table 2, the compressed file size is significantly smaller for the 3D case compared to simple Part 1 JPEG 2000 compression. The best compression is achieved when all the components in the sequence are grouped into a single collection (the N column in the table), although the loss in compression efficiency (compared to the N column) is small even with a component collection as small as 20. In the right most column is the percent reduction in file size between Part 1 JPEG 2000 compression and Part 2 JPEG 2000 with a single component collection (of all the images in the sequence). Overall, Part 2 reduces the compressed sequence size by 15-18%, for these three sequences.

LOSSY COMPRESSION RESULTS

Each of the three sequences was compressed with loss using both Part 1 JPEG 2000 and Part 2 JPEG 2000 for 2D-only and 3D compression, respectively. Compression ratios in the range of 10:1 to 50:1 were used and the resulting compressed sequences were compared using an average peak signal to noise ratio (pSNR) metric over all the images in the sequence. pSNR is a commonly used metric for image fidelity; higher pSNR is equivalent to lower distortion in the image.

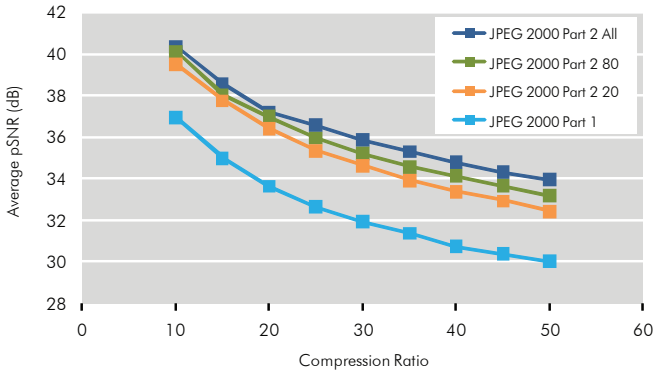

Figure 5 *- Sequence #1: average pSNR over all images in the sequence vs. compression ratio. The different Part 2 curves represent different component collections.*

To test the effect of the component collection size on compressed sequence size and resulting image pSNR, the first sequence was compressed using components collections of 20, 40, 80, and 127 (all the images in the sequence) images. The resulting average pSNRs over the entire image sequence are shown in Fig. 5, at compression ratios ranging from 10 to 50 to 1. The different Part 2 curves represent different component collections. As in the lossless case, the best results are obtained with a single component collection for the entire sequence. The average pSNR is slightly lower for the other component collections, but still considerably higher than that for Part 1 2D-only compression.

Results of compressing Sequences 2 and 3 are shown in Figures 6 and 7. These figures show the average pSNR of the resulting image sequence at various compression ratios, using a single component collection for the entire sequence. Note that the average pSNR is consistently and considerably higher than the Part 1 compression case.

Also note that at a specific pSNR level, Part 2 achieves more than twice the compression of Part 1. For example, for Sequence #2 in Figure 6, Part 1 compression results in an average pSNR of 77 dB, at 10:1 compression ratio. The same average pSNR can be achieved with Part 2 at a compression ratio of 25:1. Similarly, for Sequence 3, Part 1 achieves 67 dB at 15 to 1 compression ratio. Part 2 achieves the same pSNR value at 40 to 1 compression ratio. At higher compression ratios the gains are even larger, as seen in all three figures.

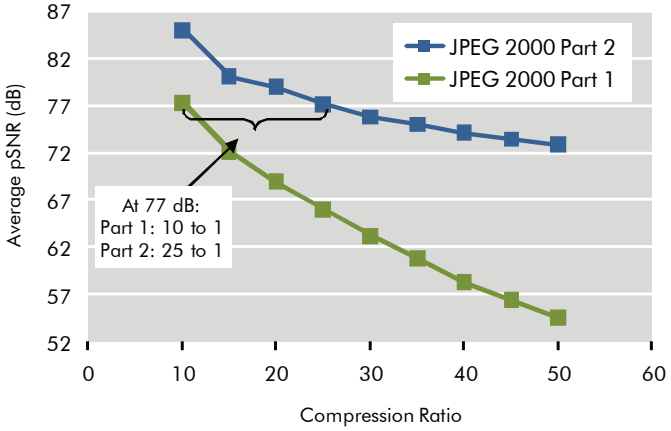

Figure 6 *- Sequence #2: Average pSNR over all images in the sequence vs. compression ratio. Note that Part 2 JPEG2000 outperforms simple Part 1 JPEG2000 by 8 to 18 dB. The Part 2 data in this figure was compressed using a single component collection for the entire sequence (429 images).*

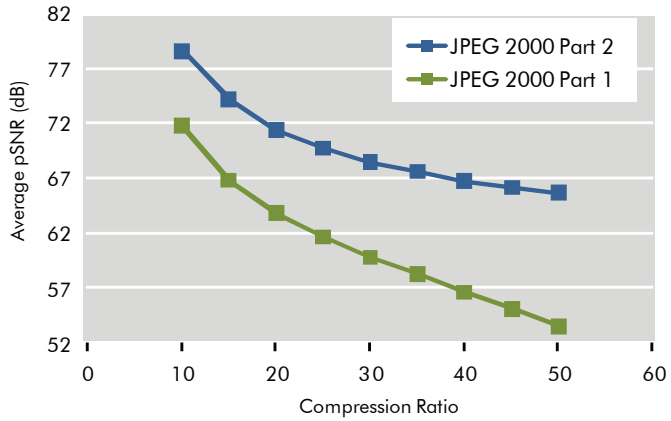

Figure 7 *- Sequence #3: Average pSNR over all images in the sequence vs. compression ratio. Note that Part 2 JPEG2000 outperforms simple Part 1 JPEG2000 by 6-13 dB. The MCT data in this figure was compressed using a single component collection for the entire sequence (620 images).*

Figure 8 shows the pSNR of each image in Sequence 2, compressed at a compression ratio of 15 to 1. Notice that Part 2 displays consistent pSNR values around 80 dB, while Part 1 achieves pSNR values of about 72 dB. Similar results can be generated at other compression ratios, as indicated by the average pSNR values in Figures 1-3.

JPIP: Standard-Compliant Streaming for Client-Server Interaction with JPEG 2000 Medical Images

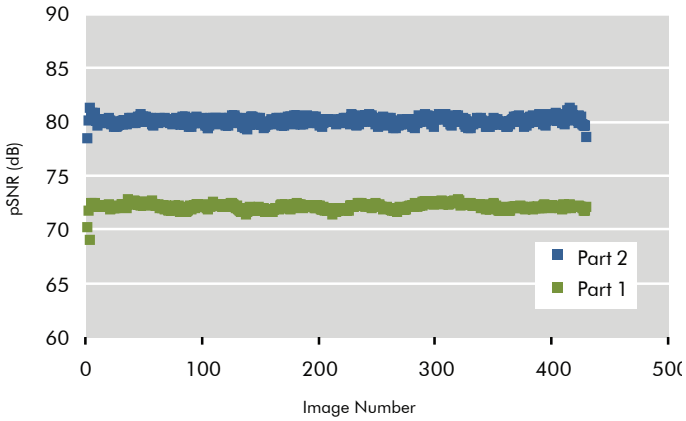

Figure 8 - Sequence #2: Comparison of the pSNR of each image in the sequence at a compression ratio of 15 to 1. Note that Part 2 JPEG2000 outperforms simple Part 1 JPEG2000 consistently by about 8 dB. The Part data in this figure was compressed using a single component collection for the entire sequence (429 images).

WHAT IS JPIP?

JPIP is a client/server communication protocol defined in Part 9 of the JPEG 2000 suite of standards, officially entitled "Interactivity Tools, APIs and Protocols". JPIP enables a server to transmit only those portions of a JPEG 2000 image that are applicable to the client's needs. Using either HTTP, HTTPs or UDP protocols, JPIP enables the client to access metadata or other contents from the image file. This capability results in a vast improvement in bandwidth efficiency and speed when performing important and valuable image viewing tasks in a client/server environment, while reducing the storage and processing requirements of the client. The larger the images—and the more constrained the bandwidth between the client and server—the greater the benefit of JPIP. In short, JPIP is an ISO standard that is designed to make optimal use of the JPEG 2000 still image compression standard to provide open, non-proprietary image streaming.

JPEG 2000 enables the extraction of subsets of a JPEG 2000 image through three standard compliant image derivation techniques: 1) spatial, 2) resolution level, and 3) quality level. That is, from a single (lossy or lossless) JPEG 2000-compressed image, a clinician can remotely view 1) a particular region of the image, 2) a large or small version of the image, or 3) a high or low quality version of the image (large or small file, respectively), or, any combination of all three. JPIP can be used to progressively stream images of increasing quality, giving the viewer at the client a view of the image as quickly as possible.

JPIP FOR MEDICAL APPLICATIONS

JPIP can be useful as a result of the following recent trends in the medical imaging community:

1. **Medical images are growing larger; often hundreds of megabytes when uncompressed.**
2. **Data output from modern scanning equipment is increasing dramatically with slice frequency.**
3. **Bandwidth is constrained; increasing demands on LAN and WAN bandwidth outpace the ability of facilities to meet them.**
4. **A centralized image storage architecture is increasingly beneficial, while demand for remote viewing and broader patient file sharing is increasing.**
5. **There is demand to make viewing stations more accessible and thus more affordable, requiring less processing power and storage.**

Take for example a medical file containing a series of 50 digital CT scanner images, each 0.5 Mbytes in size or 25 MB for the full study, uncompressed. Mostly commonly the archival storage is in RAW format. Typically, a clinician must download the whole 25 MB study to browse, zoom, pan and decide which image(s) to investigate further. This can take several seconds or even minutes, depending on the available network bandwidth. When the clinician goes to view a different image in the series, the entire image must again be downloaded.

However, the clinician can access the images dramatically faster with a system equipped with JPEG 2000 and JPIP. In this system, only a single file per image is necessary. Lower resolution thumbnails can be extracted directly out of this high-resolution JPEG 2000 "master" image and downloaded. This removes the need to store, manage, and link images of different resolutions in the database, which can be cumbersome. Once the clinician chooses to view a particular image, only the resolution layer required to view the entire image on the screen is downloaded. The quality layers are downloaded progressively to give the clinician an image as quickly as possible. When the clinician zooms into a particular region of interest in the image, only that portion of the image is downloaded,

and only the minimum resolution required. Again, the image can be downloaded progressively by quality layer. The clinician can continue to zoom into the image until the maximum resolution is reached, and pan across the image; each time downloading only the area of the image being viewed. The clinician can then scan across different images of the series, maintaining the same region of interest and resolution. Again, only the area being viewed is downloaded. The result is a dramatic increase in speed of viewing, and significant increase in the quality and efficiency of the viewing experience.

JPIP TECHNICAL OPERATION

When a clinician wants to view an image, the client application is activated. This application then accesses a local JPIP software library to create a properly formatted JPIP "request". This request contains detailed information about the specific region of the image that the clinician wishes to view, along with the desired resolution and quality layer data. The request is sent to the JPIP server via the network.

When the JPIP server receives this request, it also uses a JPIP software library to interpret and process it. To generate a response back to the client, the server application calls the JPIP library, which in turn calls the JPEG 2000 library to extract the relevant image data. This image data is packaged by the JPIP library, and then sent back to the client directly or using the application server

When the response is received by the client application, it hands it off to the JPIP library for processing. The JPIP library extracts the image data from the response, and recreates the image using the JPEG 2000 software library. The client application then uses the JPEG 2000 library to decode and display the image.

Figure 9 shows a system level block diagram of a JPIP Server and a JPIP client. On the server, the server application can be a CGI (computer graphics interface), a standalone network daemon, or can be embedded into an application server framework.

JPIP PROTOCOL STRUCTURE AND FEATURES

A JPIP-enabled client application utilizes one of two advanced streaming modes of JPIP: JPP-stream and JPT-stream. In both modes, the JPEG 2000 image file on the server is partitioned into finer elements, called "databins". These databins can be either tiles or precincts.

Databins are the basic elements of a JPEG 2000 image used by JPIP. JPEG 2000 files can be disassembled into individual databins, and then reassembled. Each databin is uniquely identified and has a very specific place within a JPEG 2000 file. Full or partial databins are transmitted from the server to the client in response to a JPIP request. The JPIP client can decode these databins and generate a partial image for display at any point while still receiving data from the server.

JPIP provides a structure and syntax for caching of databins at the client, and for communication of the contents of this cache between the client and the server. A client may wish to transmit the contents of its cache to the server with every request, or allow the server to maintain its own model of the client cache

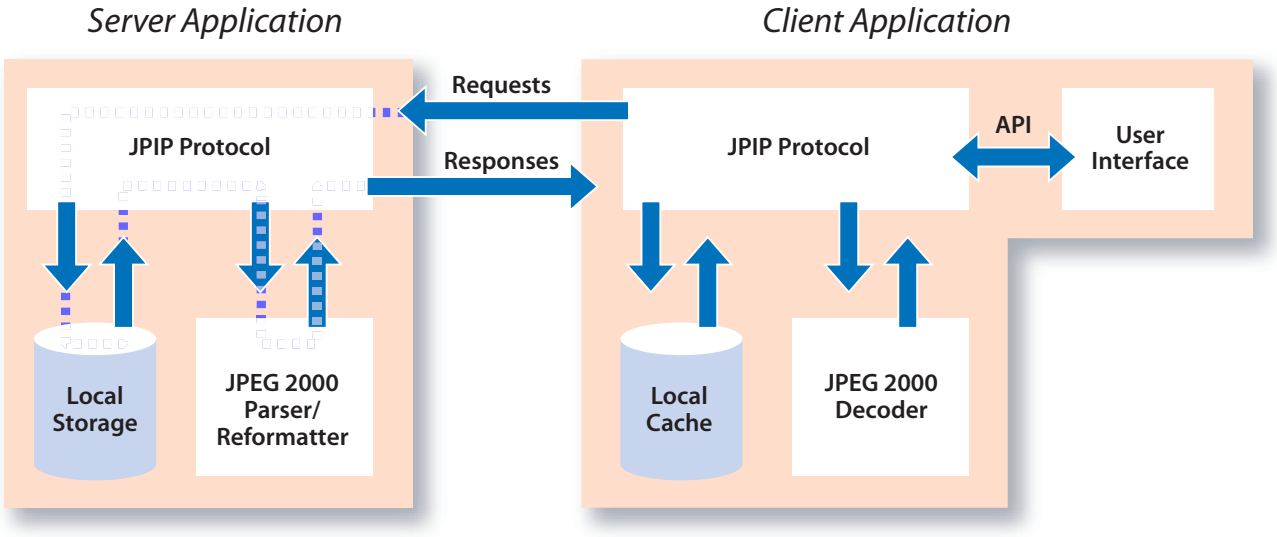

Figure 9 - *Functional architecture, JPIP client/server solution*

by maintaining a "stateful" connection. In either case, the server will reduce the amount of data it is transmitting in response to a JPIP request by eliminating the databins that the client received in previous transmissions. In this way, JPIP provides a very efficient means of browsing large image data sets in a standard-compliant fasion.

While databins are being transferred between the server and the client, they usually get split up into smaller chunks, called "messages". The JPIP server decides the JPIP message size. This flexibility to transmit partial databins enables one to vary the progressive nature of the data being sent to the client. If entire databins are sent in order, the data will be received in a progressive resolution fashion; if messages from different databins at the same resolution level are interlaced, the data will be received by the client with progressive quality. This allows applications to control the user experience depending on the application requirements.

ABOUT AWARE

HISTORY AND MARKET PRESENCE

Aware is a NASDAQ-listed company (AWRE) headquartered in Boston's technology corridor. Aware has been a leading innovator of advanced signal processing technologies since the early 1990s. Processing and analysis of digital data and imagery "signals" has vital applications in a diverse range of large and growing industries, including medical imaging and biometrics. Aware has been successful in developing products around our core signal processing expertise to address these diverse markets. While the applications of Aware's technologies are varied, there is a high degree of commonality across our product lines in terms of functionality requirements and architecture; our products perform advanced processing and analysis of digital data and images and then distribute the results in various ways throughout a client/server network.

MEDICAL IMAGING SOFTWARE PRODUCTS

Aware's medical imaging software products enable advanced image analysis, transport and viewing capabilities, including high-performance JPEG 2000 compression and decompression, radiation exposure monitoring and reporting, efficient, secure, standards-based image streaming, and zero-footprint viewing. Some of our customers have included Siemens, Merge, Philips, Cerner, CareStream and the U.S. Department of Veterans Affairs Vista Imaging Project.

FOCUS ON STANDARDS

Aware has pursued a standards based approach to its medical imaging technology. Much of this is seen in our participation in various standards bodies. This work includes:

DICOM Participation and Conformance

Co-chair position of DICOM Working Group 04 (Compression) 2009-2013

- **Author of Supplement 106 that added support for JPEG 2000 Part 2 Multi-component Transfer Syntaxes to the DICOM standard.**
- **Contributor to Supplement 105 that added support for JPIP (JPEG 2000 Interactive Protocol) Transfer Syntaxes to the DICOM standard.**
- **Served as official Liaison between DICOM WG-04 and the JPEG Committee until 2013.**
- **Closely follow Working Group 28 as an Official Observer (Physics Strategy).**

IHE Participation and Conformance

- **Co-author of the Cross-Community Access for Imaging (XCA-I) Integration Profile which describes the process of sharing patient-relevant medical imaging data between medical communities.**
- **Participating member of IHE Radiology Technical Committee.**
- **Conformance to IHE's Radiation Exposure Monitoring Profile including Image Manger/Archives and Dose Information Reporter actors, among others.**
- **Successful completion of Connect-a-thon IHE testing sessions in both Europe and North America.**

Direct, BlueButton+, and MU2 V/D/T

- **Aware is actively participating in the development of new specifications in these areas and developing solutions to meet existing specifications**

AWARE

For more information, please contact us at 781-276-4000 or sales@aware.com