Implementation Guide for the Portable Document Format in Healthcare

**Companion Implementation Guide
to AIIM Best Practice, BP-01-200X**

Prepared by AIIM

Abstract

This Implementation Guide accompanies AIIM BP-01-200X, *Portable Document Format in Healthcare: A Best Practices Guide* and provides some examples of possible implementations. This guide is designed to help facilitate implementation of technical items mentioned in the Best Practice Guide that defines a means by which information is captured, exchanged, preserved, and protected among consumers and the other participants within the healthcare system using PDF as the electronic container of the information.

Note: This document is not intended to address compliance with any applicable state and federal regulations that might apply to this information, including Public Law 104-191 (1996), the Health Insurance Portability and Accountability Act (HIPAA).

The PDF in Healthcare Work Group acknowledges ASTM International for their contributions to this Implementation Guide.

TABLE OF CONTENTS

INTRODUCTION

The migration of multiple medical record types to a universal digital format would be enabled by implementation of an easily adopted document encapsulation practice. This practice would contain specifications for portability, interoperability, and security and would promote the exchange of healthcare information.

The *Portable Document Format in Healthcare, A Best Practices Guide* (AIIM BP-01-200X) is designed as an aide to help describe the technical aspects of such a practice. This companion *Implementation Guide* is meant to be an adjunct to the *Best Practices Guide* and provides sample implementation information.

The *Portable Document Format in Healthcare, A Best Practices Guide* describes the practice of using PDF to facilitate a trusted means by which healthcare information is captured, exchanged, preserved, and protected among consumers and the rest of the healthcare system.

As an adjunct to the *Best Practices* document, this *Implementation Guide* is intended to serve as an example for implementation and does not preclude development or implementation of any other features or implementations of PDF in healthcare through additional usage models in support of storing, exchanging, sharing, or viewing of personal healthcare data.

This implementation guide recognizes that there is a continuum of implementation models ranging from large enterprise models, to independent software vendor models, small to medium sized healthcare provider offices, enterprise to physician, physician to enterprise, physician to physician, physician to other healthcare provider, patient to physician, patient to other healthcare provider, and more. This implementation guide cannot describe every use case but will provide a framework that can be replicated in a wide variety of instances.

HOW TO USE THIS GUIDE

The purpose of this guide is to provide a framework for those implementing the use of PDF as a conduit for securely moving healthcare data throughout the care universe. There are many potential implementations with dozens of use case scenarios. This guide cannot address every use case or scenario but provides a framework of basic information to assist implementers. Implementers are not limited to methods or technologies discussed in this guide. Implementers are reminded to check all state, local, and national regulations to ensure full compliance. **This document does not address any regulatory or compliance issues.**

This document is intended to be used as a reference document therefore the layout of the content does not follow a specific organizational pattern relative to implementing any of the PDF features.

IMPLEMENTATION GUIDE KIT

A package of handy tools is distributed as the *Implementation Guide Kit for PDF in Healthcare*, which can be found online at http://www.aiim.org/pdfh/ig. The samples in that kit are potential implementations that are described throughout this document. Please note all samples are for illustrative purposes only and any user name information is fictional. A fictional patient, Octavia Helios, is used in many of the physician samples. The samples included are:

CORE SAMPLES:

- **Original Doctor Files** – the files in their original format related to a use case for patient Octavia Helios.

- **PDF-Basic-converted-files** – the original physician files related to a use case for patient Octavia Helios converted to PDF.

- **PDF-ccr_xml_samples** – base PDF Files related to CCR XML and Data Enabled forms using data for patient Octavia Helios. Also includes partial *Emergency Information* form based on the AAP (American Academy of Pediatrics) and the ACEP (American College of Emergency Physicians) form.

- **PDF-Feature-Samples** – PDF files referenced throughout the document demonstrating the various features of PDF, using files from patient Octavia Helios.

- **PDF-DICOM** – PDF files with DICOM image and acquisition data.

- **PDF-DICOM_Samples** – PDF files with DICOM image and acquisition data. Described fully in the section Embedded DICOM PDF Healthcare Samples.

TEPR 2007 SAMPLES

Some additional samples of potential implementations are included in the *Implementation Guide Kit for PDF in Healthcare Supplemental 1*, also found at http://www.aiim.org/pdfh/ig:

Due to the overwhelming interest expressed by attendees of the 2007 TEPR (Toward an Electronic Patient Record) conference in Dallas, Texas, files from the PDF in Healthcare session at TEPR 2007 have been added to the *Implementation Guide Kit*. This collection of files includes Data Enabled PDF forms, a number of scanned images, various DICOM files in a PDF (CT, MR, and Ultrasound), and several XML CCR data samples.

Note that the TEPR 2007 demonstration interacted with a server. Samples of that server code are included below in the sections: Extracting and Modifying XML Data S (subsection: **XML Data Stream Extraction Sample Code**), and Inputting the XML Data Stream with XDP on a S (subsection: XML Data with the XDP File Sample Code).

Note that new additional samples may be posted to http://www.aiim.org/pdfh/ig from time to time.

FILE MANIFEST FOR IMPLEMENTATION GUIDE KIT

Table 1 lists all the files that are available in the Implementation Guide Kit, provides a general description of the file, and lists the particular PDF features for the sample file.

Table 1: Implementation Guide Kit File Manifest

Folder/File	Description	PDF Features
IG-Kit\original files		
ccr_Helios_Octavia.html	HTML generated using CCR AAFP stylesheet and **ccr_Helios_Octavia.xml** file. (AAFP stylesheet at: http://sourceforge.net/projects/ccr-resources/)	N/A
ccr_Helios_Octavia.xml	CCR compliant XML, describing Octavia Helios.	N/A
EKGheliosTES2a.wav	Dictation describing **OHekg.jpg** file.	N/A
NEJMGiantLA.gif	CT scan of the chest showing mitral regurgitation resulting in a grossly enlarged left atrium (LA).	N/A
OHekg.jpg	EKG for patient.	N/A
OHMeds.doc	A file copied from a web page with Med list for Octavia Helios.	N/A
OHMeds-scan.png	The same information as **OHMeds.doc**, but scanned from a printed hard copy.	N/A
OHreferal.rtf	Brief patient summary for Octavia Helios.	N/A
patient1.mov, patient2.mov, patient3.mov	Three embedded QuickTime movies created from source multi-frame DICOM files at full resolution.	N/A
IG-Kit\PDF-basic-converted-files		
Helios_Octavia_fromHTML.pdf	PDF file generated from **ccr_Helios_Octavia.html**.	Basic PDF
NEJMGiantLA.pdf OHekg.pdf OHMeds.pdf OHreferral.pdf	PDF files generated from similarly named files in "original files" folder.	Basic PDF
OHMeds-scan.pdf	PDF file generated from **OHMeds.png** in "original files" folder. Note this file has no text; it's just an image.	Basic PDF

Folder/File	Description	PDF Features
IG-Kit\PDF-ccr_xml_samples		
Helios_Octavia_fromHTML_attached_XML.pdf	PDF generated from **ccr_Helios_Octavia.html** with **ccr_Helios_Octavia.xml** attached.	Attachments
Octavia_CCR_PDFXMLForm.pdf	Data Enabled PDF form with internal XML capabilities using XFA. Form and **ccr_Helios_Octavia.xml** as XML data.	XML Data Enabled Form/XFA
aap_eif_form_partial.pdf	Partial *Emergency Information* form based on AAP (American Academy of Pediatrics) and ACEP (American College of Emergency Physicians) form. Form is Data Enabled and uses **ccr_Hernandez_Jane.xml** as XML Data	XML Data Enabled Form/XFA
IG-Kit\PDF-DICOM		
mammo.pdf	DICOM image and acquisition data in a Data Enabled XFA PDF form.	Data Enabled PDF with image and XML data.
Mammo_data.xml	The DICOM image and acquisition data exported from **mammo.pdf** into XML format.	Data exported from PDF
mr_attachments.pdf	DICOM image and acquisition data in a Data Enabled XFA PDF form. Original DICOM images are included as attachments.	Data Enabled PDF with image and XML data, and attachments
IG-Kit\PDF-DICOM_Samples		
See the section, Embedded DICOM PDF Healthcare Samples, for details on these samples.		
IG-Kit\PDF-FeatureSamples		
Container-attachments.pdf	A container PDF package with only attachments in the PDF file. This file includes Octavia Helios related files in one package.	Attachments
Encrypted-only-attachments.pdf	This file includes Octavia Helios related files as attachments in one encrypted PDF. To open any of the attached PDF files requires the password "pdfhealthcare" (all lower case, no quotes). However, to open the PDF itself does not require the password.	Attachments and attachment-only password encryption

Folder/File	Description	PDF Features
encrypted-password.pdf	This file includes Octavia Helios related files in one encrypted PDF package. To open the PDF file requires the password "pdfhealthcare" (all lower case, no quotes).	Attachments and password encryption
IG-bookmarks.pdf	A PDF File containing Octavia's information with bookmarks to each piece of information.	Bookmarks and annotations
IG-bookmarks-Initial-View.pdf	The file **IG-bookmarks.pdf** but with an initial view setting to open the bookmarks panel.	Bookmarks, annotations, and initial view
IG-bookmarks-Initial-View-dictation.pdf	**IG-bookmarks-Initial-View.pdf** with a bookmark which plays an audio dictation describing the ECG.	Bookmarks, initial view, annotations, and audio multimedia
IG-clinical-use-case.pdf	Similar to **IG-bookmarks-Initial-View-dictation.pdf** but with an attached **Octavia_CCR_PDFXMLForm.pdf** file and bookmark to open that attachment. Note this document is described in the Clinical Use Cases section of this document.	Bookmarks, initial view, annotations, attachments, and audio multimedia
IG-clinical-use-case_password.pdf	**IG-clinical-use-case.pdf** password protected by the clinician. Password is "pdfhealthcare" (no quotes all lower case).	Same as **IG-clinical-use-case.pdf** plus password encryption
IG-clinical-use-case_password_signed.pdf	**IG-clinical-use-case_password.pdf** signed by the clinician.	Same as **IG-clinical-use-case_password.pdf**, plus digital signatures.
NEJMGiantLA-comments.pdf	**NEJMGiantLA.pdf** with comments to explain the condition. Makes use of a "Cloud" comment to highlight area of interest.	Commenting annotations
OHekg-dictation1.pdf	ECG with audio dictation. Dictation is read when annotation is clicked.	Multimedia audio
OHekg-dictation2.pdf	ECG with audio dictation. Dictation is read when bookmark is clicked.	Multimedia audio and bookmark with play audio action

Folder/File	Description	PDF Features
OHekg-redaction.pdf	Two page file using ECG to demonstrate redaction. First page demonstrates "ad hoc" redaction; second page demonstrates use of PDF redaction feature.	Redaction
OHekg-watermark.pdf	Two page file using ECG to demonstrate watermarks. First page demonstrates original ECG; second page demonstrates use of PDF watermark feature.	Watermarks
OHMeds-scan-OCR-full-textandgraphics.pdf	**OHMeds-scan.pdf** file fully converted to text and graphics using OCR. Note file size is smaller than original, but fidelity is not the same.	Fully converted PDF
OHMeds-scan-OCR-searchable-image.pdf	**OHMeds-scan.pdf** file converted to a searchable image using OCR. Note similar file size to original, and note also that the fidelity is the same as the original.	Searchable Image PDF
OHreferral_convert_advanced.pdf	**OHreferral.rtf** file converted using a more advanced PDF generator. Note that the PDF has tags and maintained metadata from the original file, such as Author and Document Title.	Advanced PDF generated: tags and metadata
OHreferral_convert_basic.pdf	**OHreferral.rtf** file converted using a more basic printer driver based PDF generator. Note that the PDF does not have tags, and that metadata from the original file was not maintained.	Basic PDF generated: no tags or metadata
OHreferral_signed1.pdf **OHreferral_signed2.pdf** **OHreferral_signed3.pdf**	Signed PDF files using the use case. **OHreferral_signed1.pdf** has two pages. **OHreferral_signed2.pdf** adds two more pages and signs the new document. **OHreferral_signed3.pdf** adds another page and doesn't sign it. PDF tools can view the document as it was when it was signed. Some tools also provide diffing capabilities.	PDF digital signatures which allow for audit trails
OHreferral_CDS_nochangesallowed.pdf	Certified PDF File allowing no further changes to the document.	Certifying PDF signature

Folder/File	Description	PDF Features
Ultrasound_heart.pdf	Contains three embedded QuickTime movies created from source multi-frame DICOM files at full resolution.	Multimedia video. (requires QT 7)

PDF FEATURES

PDF CONTENT

This section reiterates topics presented in the *Best Practices Guide* but provides some additional content.

PDF VERSIONS AND FEATURES

In general, new features added to newer versions of PDF are not regarded as incompatibilities. Therefore, it is important to understand the native PDF features available in the various, major versions of PDF.

The *PDF Reference* version 1.5 addresses the issue of features across versions. This guide focuses primarily on *PDF Reference* version 1.7. But because *PDF in Healthcare* is a Best Practices Guide and not a standard, thoughtful consideration should be given to the version of PDF implemented relative to the native feature set.

Points to Consider: Implementers should evaluate the efficacy of keeping PDF readers and PDF tools up to date.

The file **OHMeds-scan.pdf** is converted to the file **OHMeds-scan-OCR-full-textandgraphics.pdf**, and to the file **OHMeds-scan-OCR-searchable-image.pdf**.

The file "**OHreferral_convert_advanced.pdf**" was created using a more advanced PDF generator. It generated Tags for accessibility and maintained Document Metadata information. The file **OHreferral_convert_basic.pdf** was generated using a more basic Printer driver based PDF generator. This file does not have tags, and metadata from the original file was not maintained.

OHMeds-scan-OCR-searchable-image.pdf is OCR scanned using a searchable image setting where the text is invisibly overlaid on the underlying image. This file is slightly larger than the original, but the fidelity is exactly maintained.

OHMeds-scan-OCR-full-textandgraphics.pdf is OCR scanned using a full conversion to Images and Text. Here the file size is much smaller, but the fidelity of the original image is altered significantly

IMAGE FILES

Image files can be directly converted to PDF. Also they can be attached to PDF files in their original format.

Image files such as digital photos and digital diagnostic images can be converted to PDF even at the very high resolutions required by some formats. Conversion of image files to PDF is ideal in a PDF-centric environment because not only does the conversion allow users to work in a single application, but many PDF applications provide advanced image viewing functionality such as zoom, navigation, and measuring tools. Additionally, PDF files offer support for commenting and annotations. Users can mark-up images with notes, comments, arrows, lines, and shapes to indicate areas of interest or comment. It is important to consider lossless vs. lossy conversions. Some tools will convert all images to lossy JPEG.

Points to Consider: There are many tools available to convert image files to PDF, but users should be aware of their image resolution requirements (which may be quite high for medical images) and be sure that their conversion tools support the appropriate resolutions.

Implementers should consider whether to preserve the original resolution of the bitmapped files and the implications of lossy format conversions.

When DICOM image files are converted to user viewable in-line graphics within a PDF file, there will normally be some form of non-reversible transformation required to map images from their original form into JPEG or another embeddable graphic format. This conversion is required because DICOM image files from CT, MR, nuclear medicine, X-Ray angiographic data, and computed radiographs contain pixel data stored with 16-bits/pixel of grayscale data, which is beyond the normal representation of a converted RGB value. For this reason, if diagnostic interpretation is required from a PDF, the original DICOM image files should also be available as full resolution file attachments that can be exported for review within a diagnostic quality DICOM image viewing application.

Examples:

The file **PDF-DICOM\mammo.pdf** demonstrates a file with DICOM image and acquisition data. This file is an XFA based Data Enabled PDF document. Exporting the data from this form results in the file **PDF-DICOM\Mammo_data.pdf,** which contains both the DICOM image and acquisition data in XML.

The file **NEJMGiantLA.pdf** is an image imported into PDF.

VECTOR GRAPHIC FILES

PDF directly supports vector graphic files. Unlike bitmapped (or raster) graphic files, which describe fixed-size images by the color of each individual pixel, vector graphic files are described by coordinates and mathematical formulas for lines, shapes, positions, and colors. Vector graphics may be scaled or re-sized without the loss of image quality that may occur when scaling bitmapped graphics. Vector graphics are used for graphs, charts, and waveforms. They are also used in files which describe maps, architectural plans, engineering blueprints, and other complex diagrams.

Note: The loss of raster image quality with scaling is a complex topic. It is outside the scope of this guide to address this issue.

Integrating the Healthcare Enterprise (IHE) defines PDF as the required format for electrocardiogram (ECG) waveforms and requires that these waveforms be generated using vector graphics, not images. Because vector graphics are directly supported in PDF, users are able to take advantage of the host of PDF and PDF application capabilities such as commenting, zoom, navigation, and measuring tools.

Points to Consider: *When a source document contains vector graphics, those vector graphics should generally be retained during conversion to PDF. There are tools available that can convert many types of vector graphics to PDF. Users should ensure that these tools output the vector graphics in PDF and that they do not rasterize (convert to bitmap) these graphics when producing the PDF. Vector graphics should be used where precision, scaling, and/or measurement are important, and when aliasing effects are undesirable. In many cases, vector graphics tend to be smaller than their rasterized bitmap equivalents. Therefore, when space is a consideration, vector graphics should be retained. In general, vector graphics are not suitable for photos or photo-realistic images.*

METADATA

Metadata in PDF is used to describe a whole document. Information that is usually contained in metadata includes concepts such as a document's author, creation/modification date, subject, title, and keywords.

PDF supports a very flexible format for metadata known as XMP (eXtensible Metadata Platform). XMP is not exclusive to PDF; it is based on open standard—RDF XML—as well as metadata vocabulary standards like Dublin Core (DCMI). XMP is also used as a common metadata format in other file formats. Information like document identification numbers may be stored in XMP, though it is not intended to store information such as form data. For encrypted or protected files, metadata may be either openly accessible or itself encrypted, depending on encryption parameters.

Points to Consider: *Consider the implications of storing confidential or personal information in metadata/XMP/document properties. Non-encrypted metadata can be used to store general information about documents that may be helpful for search and retrieval. If encryption is used, settings that encrypt metadata will prevent proper indexing by search engines.*

Example:

The file **OHreferral_convert_advanced.pdf** was created using a more advanced PDF generator which kept metadata information from the initial file.

ACCESSIBILITY, TAGS, AND LOGICAL STRUCTURE

PDF has features that facilitate access for visually and hearing impaired users, e.g., captioning of multimedia. This capability, often referred to as Tagged PDF, provides access to the content and logical structure of a document for use by assistive technologies such as screen readers and Braille printers. Some PDF viewers also include a built-in capability to read documents aloud.

In addition to support for assistive technology, the use of Tagged PDF also provides some benefits for all users. Tagged PDF can be re-flowed to provide easier reading for small screen devices such as PDAs or cell phones, and can be used to temporarily re-format multi-column documents into a single continuous format. The use of tagged PDF may be required to fully comply with mandates like Section 508 of the Rehabilitation Act, the ADA (Americans with Disabilities Act), or the W3C Accessibility Guidelines (WCAG).

Many PDF generating applications have the capability to create tagged PDF automatically from electronically-originated content or electronic forms. Some PDF generating applications also allow tags to be added to image-based PDF files that have been processed with OCR software. Tagged PDF can increase file size.

Points to Consider: *Consider use of PDF software that supports tagged PDF and create tagged PDF from electronic originals whenever possible. The use of tagged PDF may be impractical for image only PDF files without OCR text or for files with severe size constraints, but the implications of using inaccessible content should be considered, especially for all public-facing PDFs.*

Examples:

The file **OHreferral_convert_advanced.pdf** has tags for accessibility.

The file **OHreferral_convert_basic.pdf** does not have tags.

ANNOTATIONS

A PDF Annotation associates an object with a PDF page. Annotations are often added after the PDF has been created. PDF defines a variety of standard annotation types such as comments, links, sounds, and movies. For a full list of annotation types see *PDF Reference*, 8.4.5.

One clinical use case may involve using annotations to point out a pathological structure when a specialist forwards records to a referring physician.

Annotation Type: Comments and Markup

There are many types of comment and markup annotations. Some are in-line with document text, such as highlights, strikethroughs, and edits, while others—such as lines, notes, callouts, clouds, or shapes—exist anywhere on the page. Comment annotations may have text notes attached to them that, depending on the PDF viewing application, become visible when a user hovers or clicks on the associated annotation.

Comments can be used in a review workflow, where multiple people would like to review and comment on a document without modifying the original content. Comments allow for replies, so commenting threads can be created. PDF viewers may provide different printing options for comments.

Points to Consider: *Users can use comments for adding information to a PDF document without modifying the original content of the PDF page. When it is not desirable to change the original content of a PDF document, use comments to add information to the existing PDF.*

Use of annotations should be avoided for storing information that has form fields provided; e.g., reactions to medications should not be stored as an annotation to a medication entry.

Annotation Type: Links

Links in PDF, also known as link annotations, are much like hyperlinks in an HTML page. Links may be directly associated with text, or may be in the form of a visible or invisible box. Clicking a link in a PDF viewing application will generally redirect the user's view to the target of the link. Note that links are similar to bookmarks (see the Bookmarks section in this document) in that they facilitate navigation throughout a PDF document, to other PDF documents, or to locations on the Internet/Web.

Points to Consider: *Some PDF converters allow for the automatic creation of links from existing hyperlinks in the native document format. Some PDF tools can also add links after the PDF has already been created. Links should be used where the author would like to facilitate navigation of a document by providing hyperlinks within the document's page.*

Links to external data (via the Web) or other documents that require specific applications could impede the host document's use in a disconnected environment.

MULTIMEDIA

Various forms of multimedia (audio and video) may be embedded in PDF files or stored remotely and linked to annotations. The playback of embedded multimedia is generally dependant on a compatible media viewer being present on the viewing computer. Both audio and video may be controlled by the PDF annotation, and video may be displayed directly inside a PDF document. A

PDF file may contain multiple renditions of a single multimedia object to accommodate different formats and bandwidth constraints.

There are benefits to including multimedia files in PDFs—primarily the ability to create compound documents that let users work in a single application. Other possibilities include interactive instructional manuals or comprehensive reports that include embedded video.

Multimedia in PDF files also benefit from PDF's robust security features: if the PDF is encrypted, any media content embedded in that PDF will also be protected, and, if the PDF is digitally signed, any media content embedded in that PDF will be included as part of the signed content. One such clinical use case of multimedia in a PDF could include a video of a cardiology study in a physician's referral package.

Points to Consider: *Annotations of most types are suitable for use in many PDF applications, though they are not used for forms-type data or core document content. Care should be used in the distribution of PDFs with annotations in that they may not be fully supported by all PDF viewers. Multimedia annotations may also have dependencies on external media players that may not be present on all computers. Links are generally suitable for all documents other than those with strict archival requirements.*

Comments are generally used for collaboration, and their contents may be extracted from a PDF in data format. Visual comments may be especially beneficial for medical images or other image-type documents. Embedded multimedia should be used with caution unless all interested parties are known to have access to the appropriate multimedia player(s). The use of remote multimedia can be very beneficial with regard to file size, but requires additional care to maintain the availability of the remote content over the usable life of the document.

Examples:

PDF-Feature-Samples\OHekg-dictation1.pdf provides an area on the PDF where you can click to hear the dictation.

OHekg-dictation2.pdf provides a bookmark, which when clicked will provide the dictation.

IG-bookmarks-Initial-View-dictation.pdf provides a bookmark, which when clicked will go to the appropriate page and then provide the dictation.

PDF-Feature-Samples\Ultrasound_heart.pdf contains three embedded QuickTime movies created from source multi-frame DICOM files at full resolution.

3D

PDF includes the ability to embed collections of three dimensional objects, such as those created by Computer Aided Design (CAD) software. The 3D facilities in PDF are intended to allow authors to transmit complex 3D information such as engineering diagrams in a ubiquitous manner. These embedded 3D collections are referred to as 3D artwork, which can be included within a PDF page. Individual views of the rendered 3D artwork may also be printed. This same technology may be used in a clinical context to transmit 3D medical images.

A PDF file may include multiple instances of 3D artwork and may also include specific labeled views of that artwork. Views are based on the relationship between the camera and the 3D artwork. Some PDF viewers may provide the ability to rotate and move the 3D artwork, which allows the user to examine that artwork from any angle or position.

JavaScripts can also be included to manipulate the 3D artwork in some PDF viewers, including the animation, separation, or showing and hiding of the individual components that make up the 3D artwork. (See the discussion on use of JavaScript in the Scripting section of this document.)

Since this 3D information is contained within a PDF, those documents can take advantage of many of PDF's other features and attributes including security and electronic signatures.

Points to Consider: Implementers should ensure that all targeted viewers support 3D. The inclusion of 3D in PDF has allowed users who previously had to transmit many 2D copies or slices of a model to send a single PDF. This could be used to transmit images from sophisticated modalities such as 3D CT, MRI, etc. 3D in PDF supports the standard U3D format.

ATTACHMENTS

Any type of file can be embedded in a PDF as an attachment. Attachments can be other PDF files, other document types, multimedia, images—any file type. Attachments may or may not be associated with an annotation that is visually present on a page in a PDF document. Some PDF viewers do restrict the embedding of executable attachments for security purposes. When PDF files are digitally signed or encrypted, all attachments are included in the process.

A PDF signature verifies the integrity of all attachments, and an encrypted PDF restricts access to attachments in accordance with the settings of the PDF container. PDF does have the capability to encrypt only attachments, allowing it to function as a "secure envelope."

Points to Consider: While many file types can be attached to a PDF, PDF is the preferred format for several reasons. Using only PDF attachments helps ensure that the user will have access to all embedded content. Otherwise, a user may not be able to view a particular embedded file type without the originating application. Another benefit is that PDF attachments can be searched from within the container PDF, allowing a user to perform a single search across multiple files. PDF attachments can employ compartmentalized security—allowing each attachment to have individual security settings, encryption levels, or digital signatures.

One main use case for non-PDF attachments is to embed an original file inside of the PDF that was created from it. This allows for the preservation of the original file, while providing a universally viewable PDF. Otherwise, non-PDF attachments should follow the same rule as multimedia annotations—they should be avoided in favor of PDF versions if there is no guarantee that all intended users of the document will have access to the appropriate viewing software.

For Example:

\PDF-Feature-Samples\container-attachments.pdf demonstrates attaching PDF files in another PDF file.

FONTS

In PDF, fonts can be embedded in a document if the positioning and look of the document is important. Most PDF viewers will attempt to match the intent if the font is not embedded. **However, PDF viewers cannot guarantee complete fidelity if the font is not embedded. Depending on the font used, not embedding fonts may even render text unreadable in PDF viewers without access to the original font.** Embedding can cause an increase in the file size which will vary depending upon the complexity of the font.

Some situations, particularly those that rely on the base 14 fonts, may not necessitate embedding. Full embedding is appropriate for forms, while subset embedding is appropriate for final form documents. Non-embedded fonts for base 14 can be used in certain situations.

Points to Consider: *The font subsetting option may be used if the document will no longer be edited. Subsetting should not be used if it is intended to be an interactive form. Fonts can be embedded as per PDF Reference 1.6 and ISO/WD 19005-2. This helps to ensure a self-contained record that better preserves future rendering by offering a minimal recommended practice.* ***The repercussions of not embedding fonts should be carefully considered, especially if the fonts are symbol or non-standard fonts.***

BOOKMARKS

Bookmarks are not visible on a page; rather they are usually displayed in a separate "Bookmarks" tab in a PDF viewing application. Bookmarks may be logically nested and may be used to replicate a document's table of contents. Note that bookmarks are similar to links (described in the Annotation Type: Links section of this document).

Points to Consider: *Some PDF converters allow for the automatic creation of bookmarks from a table of contents. Some PDF tools can be used to add bookmarks once the PDF has already been created. Bookmarks should be used where the author would like to facilitate navigation of a document by providing an overall outline. This feature can be used to help users navigate through extensive medical records in the same PDF file.*

Example:

The **IG-bookmarks.pdf** document contains four elements of use case. Bookmarks exist to each of the four sections.

LAYERS

PDF documents may include layers which are sections of content in a PDF document that can be selectively viewed or hidden by authors or consumers. These may be captured from original files during generation. Layers can be used to support multiple languages. They enable a single PDF with multiple languages available for viewing.

Layers can be used to preserve a greater degree of detail and functionality for some types of content. For example, images may contain multiple layers that are hidden at lower magnification levels for clarity, but become visible at higher magnification levels to provide detail. Layers are most valuable for detailed visual content, but may not be fully supported by all PDF viewers.

Points to Consider: *Consider avoiding the use of dynamic layers to convey critical content except in situations where all users can be expected to use software that fully supports them.*

ACTIONS

PDF specifies a number of predefined *Action* types for interactive documents. Actions can include: running JavaScript, moving to a particular destination in the current document (like a hyperlink), playing a sound or movie, hiding, or showing an annotation or comment, submitting a form, or moving to a particular 3D view.

PDF Viewers which support Actions will carry them out as a result of a trigger event. PDF defines a number of trigger events. Some trigger events include opening a PDF page, activating a field,

typing into or clicking on a field, before and after saving or printing a document, among others. A full list of supported Actions and Triggers is documented in *PDF Reference*, section 8.5.

Points to Consider: *Actions and triggers can be used to provide a consistent initial view for users. When doing so, implementers should confirm that targeted PDF viewers support Actions and Triggers.*

SCRIPTING

PDF files may contain JavaScript. JavaScript can be used to make PDF documents more interactive, it can also be used to manipulate documents, customize the viewing environment, perform calculations or other actions on form fields, send or retrieve information from Web services, or provide detailed information about documents.

Implementers of JavaScript are encouraged to follow organizational security guidelines commensurate with the use of active content and scripting in their environment. This guide does not make a recommendation either way; rather, implementers should balance the effectiveness of using this technology in regard to the benefits derived versus the potential risks.

It is worthwhile to note that JavaScript in PDF is not the same as JavaScript in the browser. The security aspects are much different. In PDF, JavaScript can be used to provide increased interactivity and can be authenticated and protected using digital signatures. Implementers are reminded that scripting has value in some situations but it may not be appropriate for all documents.

Points to Consider: *Scripting can be very valuable in PDF, provided that certain considerations are taken. Since not all PDF viewers support JavaScript, implementers should ensure that PDF documents that depend on JavaScript for critical content or functionality will be used only with PDF viewers that support it.*

REDACTION

Redaction is the process of removing personal or confidential information from a document, usually prior to generalized distribution or publication. The redaction of paper documents generally involves crossing out words or sections of text with a wide black pen.

Redaction of electronic documents is somewhat more complicated, and when performed incorrectly can result in the inadvertent disclosure of the information that was presumed to be removed.

Electronic redaction in PDF must be performed correctly to avoid inadvertent disclosure of information that should be "blacked out." It is not sufficient to highlight text in a PDF or source document in black to obscure the text below. This method does not remove the actual text, which may still be extracted or exported. Some PDF tools have built in support for actual redaction. One example of redaction in a clinical use case could be to redact a patient's identity information when sending a teaching file to another institution.

Points to Consider: *Redaction of PDF documents should only be performed with tools specifically designed for it. Failure to use redaction-specific PDF tools may result in the exposure of confidential information.*

Example:

The first page of **OHekg-redaction.pdf** demonstrates how a physician may black out a patient's identity. The second Page uses the PDF Redaction tool to ensure the name is completely removed from the document or image.

WATERMARKS

Some PDF generation applications can create PDF documents with watermarks. Watermarks can appear as page headers, footers, or behind the body of a document. Watermarks may be fully visible or may be invisible on-screen and visible only on printed copies of the document.

Points to Consider: Watermarks can be useful for discouraging unauthorized use of files. They can also be useful to denote additional information about a document such as its status or origin (e.g., Draft, Confidential, or Complete).

Example:

The first page of **OHekg-watermark.pdf** demonstrates the original ECG. The second page uses the PDF watermark feature to indicate it is a teaching file.

READER CONFIGURATION ISSUES

Initial View

During the creation and/or post configuration of a PDF file, it is best to ensure the file has a consistent initial view upon opening. If no option is selected the default saved view is used, which may lead to variability between users. More importantly, offering users quick access to PDF stored and structured content on initial open is a simple and welcomed solution.

Points to Consider: Actions and Triggers can allow a document to be more interactive and can provide for navigation and help. Users hoping to provide for interactive features without the desire to write JavaScript may want to examine Actions and Triggers. Implementers should ensure that all targeted PDF viewers support Actions and Triggers.

Example:

The **IG-bookmarks-Initial-View.pdf** file demonstrates initial view, set to show the bookmarks panel when the document is opened.

FORMS

PDF documents support electronic forms. Electronic forms can be used interactively to collect information from users, and/or as templates to allow the merging of forms and data for display to users. In some processes, these features are combined to allow for the partial pre-population of forms before they are completed by users. PDF forms are used in many workflows, from small, decentralized processes in which users send and receive forms via email, to large enterprise systems that integrate with services and databases. PDF forms can be divided into two distinct types: AcroForms and XFA (XML Forms Architecture).

ACROFORMS

AcroForms is the legacy model for PDF forms. AcroForms was largely superseded by the introduction of XFA in PDF 1.5, but remains supported by many PDF viewers and is sometimes still appropriate for simple forms. AcroForms is based on simple form field annotations that are typically created within a PDF-based application. Form field annotations can be text boxes, buttons, lists, or other basic form objects. Some PDF viewers or applications allow data to be imported, collected, saved, or exported from AcroForms PDF forms.

AcroForms allows the import or export of form data in a very basic type of flat data file that may be serialized as delimited text or FDF (Forms Data Format) or XFDF (XML-based FDF). Unlike XFA, the data extracted from AcroForms-based documents generally needs to be transformed before it can comply with any industry data standards or services.

Points to Consider: *AcroForms may have a role in simple, ad-hoc, and small office use. They may also be suitable for use in an environment where compatibility with older viewers is required. They are generally not intended for use with XML standards, Web services, or enterprise systems. Not all viewers support XFA. Implementers should examine their environment for suitability.*

XML FORMS ARCHITECTURE (XFA)

XFA was added to the PDF reference 1.5 as an alternative for AcroForms. The location of the XFA Specification is included in the Bibliography section of this document. XFA forms can be anything from simple static PDF forms that mirror their paper counterparts, to highly interactive and dynamic forms with flowable content and direct support for industry XML standards. One of the main benefits of XFA is its support for direct integration with XML standards like the CCR. Integration with XML standards means that the visual elements of XFA forms can be mapped to the data structures of an XML document.

From an XFA perspective, mapping individual documents is implementation-specific and a samples are included in this Implementation Guide and the associated Kit. Mapping need not be technology specific, rather standard layouts can be mapped to XML data standards in an abstracted manner.

This mapping allows the XFA/PDF forms to provide both human-readability through PDF and machine-readability through XML. XFA-based PDF forms that are mapped to an XML schema can support the import or export of XML that conforms to the schema, and can be used to validate interactive user input against the rules and data types of that schema. XFA-based PDF forms can also be designed to be dynamic, with flowable content based on user interaction or the contents of a dataset.

Points to Consider: XFA forms provide a much greater level of functionality over AcroForms and should be used in nearly all situations where PDF 1.5 or later is in use. XFA forms may contain scripts and dynamic content, and should follow the same points to consider for scripting as in other PDF documents. Ensure use of a compatible viewer.

Example:

This document describes XFA Forms and delivers several samples. See sections: "Data Enabled PDF Healthcare Samples", "DataEnabled PDF Generation Guidelines", and "Embedded DICOM PDF Healthcare Samples"

BARCODES

PDF supports the use of barcodes, as images or barcode fonts. AcroForms support simple one-dimensional (1-D) and two-dimensional (2-D) barcodes. XFA forms in PDF support the inclusion of dynamic 1D and 2D barcodes. Barcodes encode information in a line and block-based image format designed to be printed and then reliably read by hardware scanners. Barcodes may consist of a series of bars of varying widths or points or glyphs on a grid.

1-D barcodes have one line of bars, whereas 2-D barcodes may have multiple lines or may rely on grid-type systems. As a result, 2-D barcodes can generally encode a greater quantity and/or more varied information than 1-D barcodes. There are many different types of barcodes. Some are specified formally by standards organizations and others are nothing more than conventions. XFA specifies identifiers for some of the most commonly used barcodes, while AcroForms only allows for 3 types.

Some PDF viewers may allow 1-D barcodes to accept typed user information, much as a text field does. The user can click on the barcode, type a value, and when they commit that value, the field will display the value encoded as a barcode. 2-D barcodes are usually not directly interactive, and have their value set by script or calculation based on a collection or series of other field values. All barcodes include some level of redundancy to support error correction when scanning.

Some barcodes allow the amount of error correction redundancy to be explicitly specified, since more error correction means more space on the page. Some barcodes allow data to be compressed, which can reduce the size of the barcode on the page for large amounts of data.

Points to Consider: Barcodes can be used to reliably link paper documents to electronic workflow. Users can print or fax a PDF form that contains a barcode. That printed or faxed paper form can then be transmitted to another party, who can then scan in the barcode and obtain the information back into electronic format.

Healthcare includes many workflows which rely on paper and fax. Barcodes provide the ability to incorporate these paper workflows into more accurate and automated electronic workflows. Simple, static barcodes place few if any requirements on an end user. Implementers of more complex 2-D or dynamic barcode based systems should ensure that all PDF viewers support the required fonts and forms capabilities needed in these scenarios. Barcodes can contain a limited amount of data. It is recommended that that data be compressed when attempting to encode large amounts of data.

PARTICIPANT-BASED SCENARIOS

Table 2 is a representative sample (not a complete listing) of some use cases for transporting PDF in the healthcare ecosystem. The transport of these documents can occur in a variety of methods depending upon each implementer's security protocols and procedures. Some examples could be using secured email, using a removable USB drive, using a CD or DVD, creating a secure PDF document, etc.

Table 2: Sample Use Cases for Transporting PDF

Scenario	Description
Enterprise to Enterprise	Large healthcare institution sends records to another healthcare Institution
Clinic to Clinic	Clinic or medium sized practice sends records to clinic or medium sized practice
Clinic to Enterprise	Clinic or medium sized practice sends records to enterprise
Employer to Patient	Employer health and wellness program sends medical records to patient/employee
Enterprise to Clinic	Large healthcare institution sends records to private clinic or practice
ISV	Independent software vendor creates medical record data from medical provider software
PACS System to Enterprise/Clinic/Provider/Patient	PACS system sends digital image to any downstream recipient using secured PDF container
Patient Record to Emergency Responder	Emergency responder accesses patient's Emergency PDF health record
Patient to Educational Facility	Patient sends child's immunization records to school
Patient to Employer	Patient sends medical records to employer health and wellness program
Patient to Insurer	Patient sends medical records to insurer
Patient to Personal Health Record (PHR)	Patient sends selected health care data to their PHR
Patient to Physician	Patient sends medical records to physician or other healthcare provider
PHR to Patient	PHR sends medical records to patient
PHR to Recipient in Healthcare Universe	PHR sends medical records to any approved recipient in care universe
Physician to Physician or Provider to Provider	Physician or healthcare provider sends medical records to another Physician or Provider

CLINICAL USE CASES

This use case describes one possible scenario to add image files to a CCR based PDF.

Use Case: Adding Image Files to a CCR-based PDF

Mrs. H visits her cardiologist because of 3 months of coughing blood accompanied by shortness of breath on exertion and rapid heart rate. She has a history of a heart valve replacement and takes an anti-coagulant medication. She has a number of other chronic medical conditions including diabetes. Based on his evaluation, the cardiologist elects to refer Mrs. H to a cardiac surgeon for further evaluation.

After the visit is completed and some basic test results are back (including an ECG and a chest CT scan), the cardiologist generates a CCR summary from the electronic health record system and transforms the CCR into PDF. The clinician creates a PDF from the referral letter, and images of the ECG and the CT scan. The clinician adds a dictation to the ECG, and adds some comment annotations to the CT scan. The clinician then attaches the PDF with the CCR to this PDF, and adds some bookmarks to ease navigation throughout the document. (This file is **IG-clinical-use-case.pdf**.)

At this point the document is password protected by the clinician and then signed. (These files are **IG-clinical-use-case_password.pdf** and **IG-clinical-use-case_password.pdf_signed**, respectively)

Once completed, the PDF is included as an attachment to a secure email that is sent on to the referral surgeon. The physician may also send the attachment in secure emails to Mrs. H's primary care physician and to Mrs. H herself for her personal health record. Figures 1-3 illustrate the information contained in the patient's PDF file.

Brief Patient Summary/ Octavia Helios DOB 1/2/45   MRN# A212222

As of 2/2/07

#1 Current Problem: Recurrent Hemoptysis, getting progressively worse.

HPI: Mrs. H. has developed hemoptysis over the past 3 months, first just nocturnal and now daytime also. She also has noted increasing tachycardia and dyspnea. She is worried and realizes this may mean she needs her Mitral valve replaced since it has been in place since her surgery at SSMCH in April 1979.

Her history includes the resection of a Congenital Subaortic Membrane associated with angina on 6/24/77 also at SSMCH. Post op she developed recurrent tachyarrhymthias and eventually Chronic Atrial Fibrillation. She also had Rheumatic Mitral Stenosis which was treated by the MVR in 1979, just 2 years after her resection for the Membranous SAS. A Bjork Shiley #27 device was inserted and seems to have lasted a very long time.

Mrs. H. also developed NIDDM a few years ago and this has been managed by diet, metformin and glyburide.

PMH: She visits her family in France frequently and stays for extended periods. Approximately 7 years ago she experienced a small stroke for which she has no residual deficit. It is thought to have been due to an embolus, but never proven. Her warfarin management over the past 27 years has been good, though prior to the small CVA, the frequency of PT/INR checking was mildly sub-optimal. Two years ago while in France she developed CHF, ascites and possibly septicemia preceding this, though I do not have all the documentation.

There is also a history of Sciatica for which she has seen Henri Bone, MD in the past and she takes Vicodin on occasion.

SH: Married with 3 grown children including her daughter, Marie who acts as her translator (though Mrs. H. understands basic English spoken slowly). She has been a busy homemaker and mother for most of her life and has excellent family support from my perspective as her cardiologist since I first discovered her problems in late 1976.

Non smoker and rare ETOH intake.

Meds: See attached list.

Allergies/ ADRs: Lisinopril caused coughing and last summer she fainted on Carvedilol in France where it was stopped. I have just restarted the carvedilol at a very low dose to better control her AFIB rate. She has no antibiotic allergies.

Currently, her plan is as follows:

I discussed her case with Brian Santos, MD, who will be seeing her on 2/5/07 with the idea that she may require catheterization to better document her pressures and gradients. She has no evidence of CAD on a recent Myoview stress test. Brian asked me to refer her to ENT to look for alternative causes of her hemoptysis, so she will be making appointments for that soon. I also asked her to see Dr Oliver, her PCP as I recently increased her metformin to improve her glycemic control.

A CT of the chest revealed no malignancies or obvious pulmonary disease. A recent Echo on 2/21/07 shows an EF of 75% with mild anteroseptal hypokinesis, perhaps related to her original membrane surgery. There is moderate bi-atrial enlargement and the RV and aortic root are dilated. She has moderate/severe AI and trivial AS. There is moderate to severe TR with mild PAH of 41 mmHg. Her echoes, labs, CT images etc are all in the SSMCH LMR/computer. Her ECG shows AFIB – unchanged over several years.

I have scheduled a 24hr Holter to be done soon to assess her rate control.

Her PCP is Frank Oliver, MD

Her daughter Marie's contacts are (888) 999-9999-mobile and (999) 999-9999 – work.

Her home phone is (999) 999-999

Thomas E. Duitrite, MD

999 999-9999 (mobile)

999 999-9999 (office)

Figure 1: Patient Summary

Octavia Harris Med list

Medications: (Detail)   (Mini)   [Renew Selected]   [Select All]   [Select None]

- ☑ Aldactone (spironolactone) (Tablet 25 mg) 2 tablet once a day Disp. 360 Rfl #6 (last: 01/31/2007) stop on: 07/14/2008 [Renew] [Prescribe] [Stop]

- ☑ amoxicillin (Tablet 500 mg) 4 tablet single dose as directed Disp. 4 Rfl #3 (last: 01/31/2007) [Renew] [Prescribe] [Stop]

- ☑ Coumadin (warfarin) (Tablet 5 mg) 1 1/2 tablet once a day as directed Disp. 180 Rfl #3 (last: 01/31/2007) [Renew] [Prescribe] [Stop]

- ☑ Digitek (digoxin) (Tablet 250 mcg) 1 tablet once a day Disp. 90 Rfl #3 (last: 01/31/2007) [Renew] [Prescribe] [Stop]

- ☑ Diovan (valsartan) (Tablet 80 mg) 1 tablet once a day Disp. 90 Rfl #3 (last: 01/31/2007)

[Renew] [Prescribe] [Stop]

- ☑ ferrous gluconate (325 mg) ** 1 mg once a day Disp. 90 Rfl #3 (last: 01/31/2007) [Renew] [Prescribe] [Stop]

- ☑ FREESTYLE TEST STRIPS 100'S ** AS DIRECTED Disp. 100 Rfl #3 (last: 01/31/2007) [Renew] [Prescribe] [Stop]

- ☑ glyburide (Tablet 2.5 mg) 1 tablet twice a day Disp. 60 Rfl #5 (last: 02/16/2007) [Renew] [Prescribe] [Stop]

- ☑ GLYBURIDE 2.5MG TABLETS ** 1 tablet once a day Disp. 90 Rfl #3 (last: 01/31/2007) [Renew] [Prescribe] [Stop]

- ☑ Lasix (furosemide) (Tablet 40 mg) 2 tablet every morning Disp. 180 Rfl #3 (last: 01/31/2007) [Renew] [Prescribe] [Stop]

- ☑ metformin (Tablet 500 mg) 1 tablet every morning Disp. 270 Rfl #3 (last: 01/31/2007) [Renew] [Prescribe] [Stop]

- ☑ Vicodin (hydrocodone-acetaminophen) (Tablet 5-500 mg) 1 tablet twice a day as needed for pain Disp. 30 Rfl #3 (last: 01/31/2007) [Renew] [Prescribe] [Stop]

Allergies:

- Coreg (carvedilol) (fainting in France)

- Zestril (lisinopril) (cough)

Problems:

- 35.24 Replacement of Mitral Valve Not Elsewhere Classified

- 427.31 Atrial Fibrillation

- 428.0 Congestive Heart Failure Not Otherwise Specified

Figure 2: Patient's Medication List
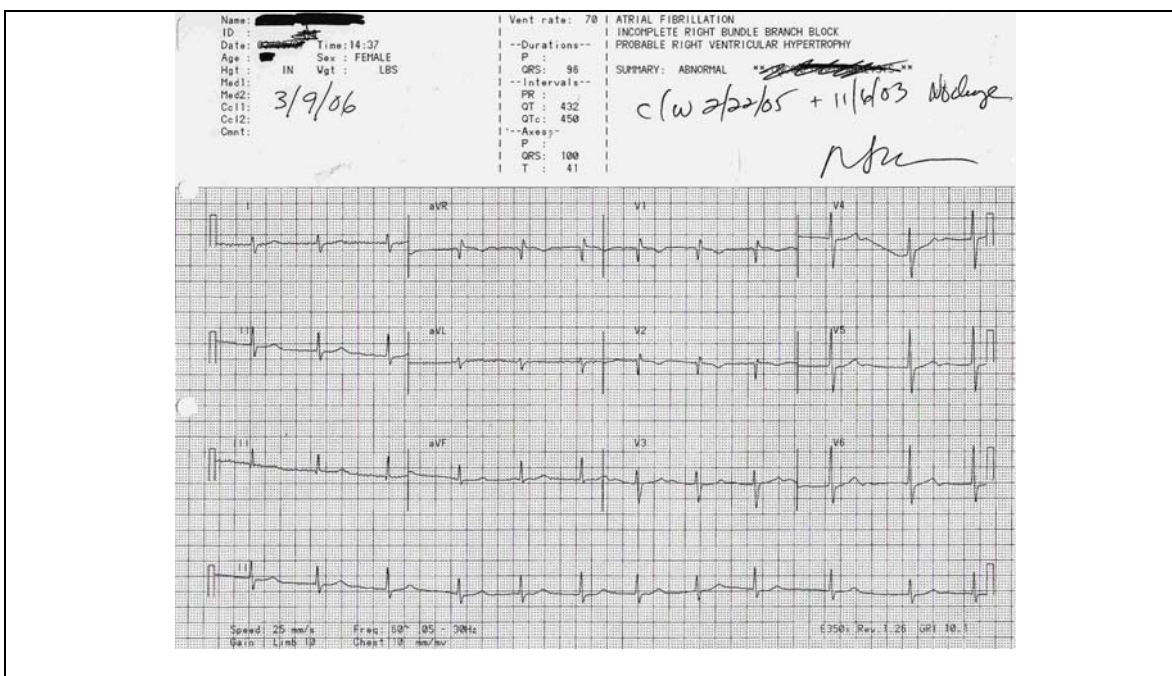
Figure 3: Patient's Test Results Image

DATA INCLUSIVE PDF HEALTHCARE SAMPLE

The data inclusive sample is generated from an HTML rendering of a CCR XML file. The HTML file was rendered using the stylesheet located at "http://sourceforge.net/projects/ccr-resources/". The original CCR XML is attached to the PDF file.

Example:

IG-Kit\PDF-ccr_xml_samples\Helios_Octavia_fromHTML_attached_XML.pdf

DATA ENABLED PDF HEALTHCARE SAMPLES

The data enabled samples were created using the following:

Tools:

- Adobe LiveCycle Designer 8.0

- Adobe Acrobat 8.0

Other:

- CCR Schema subset (**ccr_simple_subset.xsd**)

- ASTM E2369-05, *Standard Specification for Continuity of Care Record (CCR)*

References:

- *XML Forms Architecture (XFA) Specification,* Version 2.4
  <http://partners.adobe.com/public/developer/en/xml/xfa_spec_2_4.pdf>

- *Adobe XML Form Object Model Reference*
  <http://partners.adobe.com/public/developer/en/xml/Adobe_XML_Form_Object_Model_Reference.pdf>

Samples:

The **IG-Kit\PDF-ccr_xml_samples\design** folder contains the designer related files used to create the file **Octavia_CCR_PDFXMLForm.pdf**.

Table 3: Data Enabled Samples Files

File	Description
CCR_PDFXMLForm.xdp	XML Data Package (XDP) containing XFA template syntax for this form
ccr_simple_subset.xsd	The subset of the CCR used to define the form's data description and the bindings to the fields
ccr_Helios_Octavia.xml	The sample XML data used by the form

The **IG-Kit\PDF-ccr_xml_samples** folder contains the following file which is also Data Enabled.

Table 3.1: Data Enabled Samples Files

File	Description
aap_eif_form_partial.pdf	Partial *Emergency Information* form based on AAP (American Academy of Pediatrics) and ACEP (American College of Emergency Physicians) form. Form is Data Enabled .

DATA ENABLED PDF GENERATION GUIDELINES

ASSUMPTIONS

This section assumes a basic understanding of Adobe LiveCycle Designer 8.0 and Adobe Acrobat 8, or similar comparable software. These products were used to create the reference tools for the development of the sample. Implementers may use any comparable tool set. This section also assumes some experience in designing XML based forms, including Data Binding and SOM Expressions.

DATA BINDING

Data Enabled XFA based PDF files can serve as a host for XML data. They can also allow for the import and export of valid instances of any pre-configured XML instance or schema. In practice, this is accomplished by specifying both an `<xsdConnection>` element in the `<connectionSet>` section, and a `<dataDescription>` element within the XFA Dataset. The `dataDescription` element, described in detail in the *XFA Specification*, corresponds to a subset of the W3C XML Schema.

Adobe LiveCycle Designer 8.0 specifies these elements through the Data View panel. The following steps allow for the creation of this element:

- Select menu item Window => Data View.

- In the Data View menu (triangle at the top right), click on "New Data Connection".

- Select "XML Schema", then name the Data Connection **ccr_simple_subset**, and click on "Next".

- Select the XSD file **ccr_simple_subset.xsd**, which should be in the same directory. This should result in **.\ccr_simple_subset.xsd** displayed in the edit box. Under the Options dropdown "Use XML Data Root Element Name", select "ContinuityOfCareRecord", and click "Finish"

This will create the appropriate ConnectionSet and Data Description Elements. It will then present a hierarchy in the "Data View" panel corresponding to the **ccr_simple_subset.xsd** schema.

Data Binding Fields and Subforms

Fields and subforms are bound by specifying the `<bind>` element.

Attributes in the bind element include `match` and `ref`, where `match=` `none`, `once`, `dataref`, or `global`. The default is `once`.

The `ref` attribute specifies the explicit data node to match to in the case where `match=dataref`.

Adobe LiveCycle Designer 8.0 specifies the `<bind>` element when the user drags an element from the "Data View" panel into the "Body Pages" section. Dragging an element into an empty area will result in the creation of a field, or a hierarchy with subforms and/or fields, depending on what element the user drags from the "Data View" panel. Dragging an element onto an

existing field will result in the updating of the target field's bind element. The sample form's simpler bindings were specified using this mechanism.

Selecting a Subset of Sibling Nodes

In some cases the sample selects only a subset of nodes from the data XML to display. In particular, the CCR makes use of the ActorID in several places to reference an Actor in the Actors section of the CCR. XFA allows the user to specify an expression to be evaluated when selecting data nodes. If the expression evaluates to TRUE, then the node will match, otherwise it will not. Either FormCalc or JavaScript can make up the contents of the expression. This is sometimes referred to as the predicate binding technique or method.

Within Designer one could add the following expression in the Object->Binding panel for a subform:

```
$record.Actors.Actor.[ActorObjectID == $record.Patient.ActorID.value]
```

Given this expression, XFA will match all those nodes inside of Actors.Actor, where the ActorObjectID is equal to the Patient.ActorID.value expression. In the context of the CCR, this will return only one Actor node.

The XFA Specification 2.4 provides details of this mechanism under the heading "Selecting a Subset of Sibling Nodes" in Part 1 of the document.

Expressions in [] will be evaluated as FormCalc, whereas expressions in () will be evaluated as ECMA Script-357 (JavaScript).

In some scenarios a form may require a more complex data display relationship than data binding can provide. In these cases one can use a variety of scripting techniques to display and manipulate the data and form in question. The entire XFA Object Model is accessible to JavaScript and FormCalc Operations. This object model reference is available for download. Please see the "Scripting" section below for details on how the sample form uses script.

More On Data Binding

Please see the section below, entitled "More On Data Binding" for Data binding techniques used in another sample form.

TABLES

The sample makes use of Tables. XFA defines tables by using <subform> elements with a layout property of "table".

```
<subform layout="table">
```

Within the table authors can specify the content of rows and columns, and their designated layout properties such as column width. XFA's Layout engine will ensure that the layout of the specified fields conform to a tabular format.

Authors can create tables in Designer 7.1 by going to the Library panel and dragging a Table object to the Body page. Designer will then give the author options on creating the base table, which include specifying the number of columns and rows, and whether to include header and footer rows. Because a table is a subform, tables can specify XML Schema bindings, and the tables in the sample do just that. Designer allows authors to specify table bindings in the Object Panel and Binding subpanel.

XFA defines Table Rows by using `<subform>` elements with a layout property of `"row"`. These rows can also specify bindings and can contain other XFA elements such as subforms, fields, and even other tables. Each of these XFA elements contained in a row subform is considered to be a cell in the table. Because a table row is also a subform, table rows can also specify bindings, and the table rows in the sample do just that. Bindings are specified in the standard way using the Object Panel and Binding subpanel.

The sample displays a table within the cell of another table in a few cases. One case is the "Alerts" table, which includes a table in the "Agent" cell. This included table has a repeating table row element which contains a single field bound to data in the Agent section of the CCR XML Schema. All bindings are specified in the standard way described above.

SCRIPTING

The sample **IG-Kit/PDF-ccr_xml_samples/Octavia_CCR_PDFXMLForm.pdf** defines a global function, in the GlobalScripts object, by the name of `GetActorName`. This function takes `ActorID` as a parameter, and returns the name of the actor corresponding to that `ActorID` by doing a lookup in the Actors section of the CCR XML Data. The `GetActorName` method makes use of the "resolveNodes" and "resolveNode" methods provided by the XFA Object Model, to lookup the actor name. `GetActorName` is called in several cases in the sample since the CCR uses the `ActorID` throughout to identify the appropriate actor.

The `GetActorName` method is called in the initialize script of several fields. In these scripts the returned value is then assigned to the "rawValue" property of the desired field. In the tables, the "Source" cell shows the result of the `GetActorName` call. These tables actually have two fields to support this, one called '`txtSrcID`' and one called '`txtSrcName`'. The `txtSrcID` field, which is not visible to the user, uses the standard XML Data binding mechanism to get the `ActorID` related to the appropriate section in the table. For example, in the Alerts Table, the `txtSrcID` refers to the particular Alert's `Source Actor`. In order to display a more human readable format, the initialize script of the `txtSrcID` field calls `GlobalScripts.GetActorName` and displays that name in the `txtSrcName` field, which is displayed to the user.

The sample displays a table within the cell of another table, or a 'Nested Table', in a few cases. Nested Tables provide a mechanism to represent multiple values (child table) of a column in another table (parent table). Within the sample form, the "Medications" section is a parent table. Nested in the definition of "Row1" is a child table labeled "tblDirections". This relationship is illustrated in the hierarchy panel, which can be seen Figure 4.
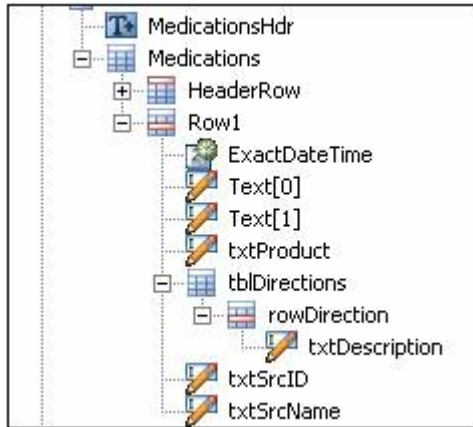
Figure 4: Hierarchy of Fields

The cardinality of the CCR XML Schema and the binding of the fields are illustrated by the Bindings Report in Designer, as illustrated in Figure 5.

Figure 5: Field Bindings

This binding report can be found in the "Report" panel, "Bindings" sub panel, by selecting the option "Fields with Data Bindings by Reference" from the dropdown.

The logical view of a section of the Medications element from the XML Schema further illustrates the binding used as shown in Figure 6.
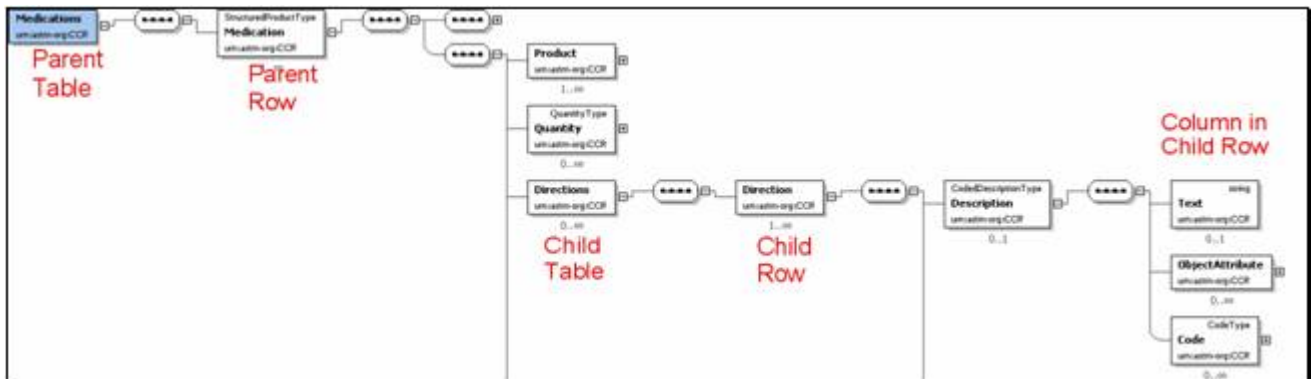
Figure 6: Logical Schema View

SCRIPTING SAMPLE

The sample defines a global function GetActorName in the GlobalScripts object. This function takes ActorID as a parameter, and returns the name of the actor corresponding to that ActorID

by doing a lookup in the Actors section of the CCR XML Data. The GetActorName method makes use of the "resolveNodes" and "resolveNode" methods provided by the XFA Object Model, to lookup the actor name as follows:

resolveNodes(STRING param), and resolveNode(STRING param) (on the tree class)

These two methods, documented in detail in the XML Form Object Model Reference, search for nodes which match the parameter entered. The parameter is the SOM (Scripting Object Model) expression to search for.

GetActorName is called in several cases in the sample since the CCR uses the ActorID throughout to identify the appropriate actor. Table 4: is an annotated version of GetActor.

Table 4: GetActor

```
function GetActorName(sActorID) {
  // Get a treelist of all Actor elements
  // In this case "resolveNodes" searches the xfa.record node for all
  // children nodes which match the SOM Expression "Actors.Actor[*]"
  // This will return a treelist with all the Actor elements under Actors.
  var oActors = xfa.record.resolveNodes("Actors.Actor[*]");
  var nNodesLength = oActors.length;
  // Loop through the Actor nodes and look for specific ID.
  for (var nNodeCount = 0; nNodeCount &lt; nNodesLength; nNodeCount ++)
  {
    // here the script looks for an Actor with ActorObjectID equal
    // to the sActorID argument
    if (oActors.item(nNodeCount).ActorObjectID.value == sActorID) {
      // once this actor node is found, the script uses resolveNode repeatedly to
      // find the appropriate Actor name to return depending on whether this Actor
      // is an Organization, Information system or Person.
      if (oActors.item(nNodeCount).resolveNode("Organization.Name") == null) {
        if (oActors.item(nNodeCount).resolveNode("InformationSystem.Name") == null)
{
          if (oActors.item(nNodeCount).resolveNode("Person.Name.DisplayName") ==
null) {
            return "No Name available";
          }
          else {
            return oActors.item(nNodeCount).Person.Name.DisplayName.value;
          }
        }
        else {
          return oActors.item(nNodeCount).InformationSystem.Name.value;
        }
      }
      else {
        return oActors.item(nNodeCount).Organization.Name.value;
      }
    }
  }
}
```

The GetActorName method is called in the initialize script of several fields. In these scripts the returned value is then assigned to the "rawValue" property of the desired field. In the tables, the "Source" cell shows the result of the GetActorName call. These tables actually have two fields to support this, one called txtSrcID and one called txtSrcName. The txtSrcID field, which is not visible to the user, uses the standard XML Data binding mechanism to get the ActorID related to the appropriate section in the table. For example, in the Alerts Table, the txtSrcID refers to the particular Alert's Source Actor. In order to display a more human readable format, the initialize

script of the txtSrcID field calls GlobalScripts.GetActorName and displays that name in the txtSrcName field, which is displayed to the user. Table 5 is an example of such an initialization script.

Table 5: Initialization Script

```
<event activity="initialize">
  <script contentType="application/x-javascript">
    // because of the binding of this field, this.rawValue will return the
    // ActorID of the source.
    if (this.rawValue != null) {
        // use the script method GetActorName to look up the name
        var sActorName = GlobalScripts.GetActorName(this.rawValue);
        // set the 'rawValue' of the field "txtSourceName" to the actor name
        this.parent.txtSourceName.rawValue = sActorName;
}
  </script>
</event>
```

The sample makes use of the instance manager. The instance manager is explained in detail in the Adobe XML Object Model reference. It is the runtime object responsible for creating, removing, and moving form model objects. The instance manager can be accessed by referring to the "instanceManager" property in a subform. The instance manager can also be accessed by prepending an underscore (_) to the name of the subform. Table 6 is a sample script that makes use of the instance manager:

Table 6: Instance Manager Script

```
<event activity="click">
  <script contentType="application/x-javascript">
    // this accesses the Address subform's instance manager by using
    // the '_' underscore shortcut.  The parameter 1 indicates that the
    // the new subform will have a corresponding data value in the data model
    _Address.addInstance(1);
  </script>
</event>
<event activity="click">
  <script contentType="application/x-javascript">_
    // this accesses the Address subform's instance manager by using
    // the '_' underscore shortcut.  The parameter "this.parent.index" passes
    // in the index of this field's parent subform as the subform to remove.
    _Address.removeInstance(this.parent.index);
  </script>
</event>
```

MORE ON DATA BINDING

The form **aap_eif_form.pdf** makes further use of the predicate binding technique described in the section Selecting a Subset of Sibling Nodes. The form uses this technique to better accommodate the CCR Schema's lookup data structure mechanism without the use of extensive scripting (like that described in the section Scripting Sample).

For the first example of this technique, see the following two elements:

1. Field: ContinuityOfCareRecord.Page1.EmergencyContactActorID

   • Binding ref is:

```
$record.Body.Support.SupportProvider.((ActorRole && ActorRole.Text) ?
ActorRole.Text.value == "Emergency Contact" : 0).ActorID
```

- In the actual xml this looks like this due to the multiple quotes:

```
<bind match="dataRef"
ref="$record.Body.Support.SupportProvider.((ActorRole &amp;&amp;
ActorRole.Text) ? ActorRole.Text.value == &quot;Emergency
Contact&quot; : 0).ActorID"/>
```

2. Subform: `ContinuityOfCareRecord.Page1.sfMainEmergencyContact`

- Binding ref is:

```
$record.Actors.Actor.[ActorObjectID==$form.ContinuityOfCareRecord.Page
1.PrimaryCarePhysicianActorID.rawValue]
```

- In the actual xml this looks like this due to the multiple quotes:

```
<bind match="dataRef"
ref="$record.Actors.Actor.[ActorObjectID==$form.ContinuityOfCareRecord
.Page1.EmergencyContactActorID.rawValue]"/>
```

Note two things.

1. The field's bind ref uses the technique described in the section on Selecting a Subset of Sibling Nodes. It uses this technique to select the nodes in `Body.Support.SupportProvider` where the `ActorRole.Text.value` is "Emergency Contact." It stores this `ActorID` in the field value.

2. The subform's bind ref also uses the technique in the above section. But, in its bind ref, it refers to the first field in the Form DOM:

```
$record.Actors.Actor.[ActorObjectID==$form.ContinuityOfCareRecord.Page1.P
rimaryCarePhysicianActorID.rawValue]
```

Care must be taken when using a bind ref expression like the second one. These expressions are evaluated during the form merge, which means that at the time of evaluation the form DOM is not fully constructed. The expression references:

```
($form.ContinuityOfCareRecord.Page1.PrimaryCarePhysicianActorID.rawValue)
may not exist at the time of evaluation, and the merge may result in an
error. The example form works because the Field
"ContinuityOfCareRecord.Page1.EmergencyContactActorID", is declared in the
template before the subform
"ContinuityOfCareRecord.Page1.sfMainEmergencyContact".
```

Because of this, when the second binding expression is evaluated, the first field will already be constructed in the Form DOM.

This example form uses this same technique with the following field and subform pairs:

```
EmergencyContactActorID, sfMainEmergencyContact

PrimaryCarePhysicianActorID, sfMainPrimaryCarePhysician

BloodPressureObjectID, sfBloodPressure
```

CONFIGURATION ITEMS

- There is no check for multiple Marital Status Entries in Social History.

- Demographics fields could be reformatted to utilize landscape form.

- The form will display existing multiple instances of Patient IDs but it needs the capability to add and delete instances as in address, telephone, and email.

- A snapshot of the form's field bindings is provided so reviewers can reference the XML Schema elements and cross reference them to the CCR documentation.

- For InstanceManager, reference the methods used in the forms, `addInstance` and `removeInstance`.

- Code snippets of how it is used are included for clarification.

- Tables: Nested tables provide support for varying levels of cardinality within an element. It may be easier to build the tables using the table tool as opposed to dragging an entire element from the data view in Designer. If the element has a gratuitous number of sub-elements, it can get quite busy on the screen and deleting tens of elements may take longer than just drawing the table from scratch.

EXTRACTING AND MODIFYING XML DATA STREAM

This section provides a quick overview and primer on accessing the XML data in a Data Enabled XFA-based PDF. It also provides sample code for use with an available commercial vendor's PDF Library. Full file format details are available in the *PDF Reference* and *XFA Specification* reference in the Bibliography.

XML Data Stream Location in the PDF File:

The XML Data in an XFA based PDF is in one stream in the PDF. This stream can be accessed using any software library or tool that provides access to the underlying structures of PDF files. XFA tools may be required, but simple extraction and modification of an XML Data Stream may not require a full XFA processing application.

A PDF contains a number of data types, which include dictionaries, streams, arrays, and strings, among others. PDF files contain a top level dictionary known as a Catalog, which links to other structures such as pages, a field hierarchy, accessibility information, and other resources. The XML Data stream for an XFA based Data Enabled PDF is found in the following location within a PDF:

Catalog => Acroform => XFA

(For full details on the internal structure of PDF please see the *PDF Reference* manual. A description of the overall syntax is in Chapter 3.)

Catalog and Acroform are both dictionaries, and XFA is usually an array (occasionally it may be a stream instead). The XFA array will reference other XFA streams. (See the *PDF Reference* section 8.6.7 in Chapter 8 for details on the exact format.) The XFA array will consist of the packets that together make up the XFA resource. (Usually these packets will include at the very least a 'template', 'datasets', and 'config' packet). The first and last elements in the array will always reference the start and end of the xdp packet. XDP is fully described in the XFA Specification.

Example of the XFA structure as a PDF array:

```
1 0 obj                          XFA entry in interactive form
dictionary
<< /XFA [(xdp:xdp) 10 0 R        XFA resource specified as individual
```

```
packets
  (template) 11 0 R
  (datasets) 12 0 R            XML Data is kept in datasets packet
  (config) 13 0 R
  (/xdp:xdp) 14 0 R ]
>>
endobj
```

The XML data is located in the (datasets) packet. In order to extract or input the XML data, it is necessary to access that packet. In the example the packet references object "12 0". In the PDF file this might look like this:

```
12 0 obj
stream
<xfa:datasets xmlns:xfa="http://www.xfa.org/schema/xfa-data/1.0/">
    ...contents of datasets packet...
</xfa:datasets>
endstream
```

A PDF toolkit which allows access to a PDF file using the basic PDF dictionary, stream, and array structures should allow access to this stream. Note that extracting and modifying this stream directly may have caveats for some documents:

- Some XFA templates may specify the application of XSLT transforms on export and import. This may not be automatically applied when extracting or modifying data directly. Not applying an XSLT where indicated may result in potential inconsistencies in data.

- Modifying data in a digitally signed file may invalidate digital signatures. Appropriate steps must be taken for the versioning of digitally signed files per the PDF Reference. Data may be extracted without modifying or invalidating a digitally signed file.

XML Data Stream Extraction Sample Code

Table 7 is a sample C# code snippet that obtains a CCR (represented in a .NET XmlDocument object) from a stream representing an XFA PDF file. This utilizes the ABCPDF.NET library (v6).

Table 7: Sample XML data stream extraction code

```csharp
using WebSupergoo.ABCpdf6;
using WebSupergoo.ABCpdf6.Objects;
using WebSupergoo.ABCpdf6.Atoms;
public XmlDocument GetCCR(Stream steam)
{
    Doc theDoc = new Doc();
    byte[] fileContents = new byte[stream.Length];
    stream.Read(fileContents, 0, fileContents.Length);
    theDoc.Read(fileContents, TextBoxPassword.Text);
    StringBuilder txt = new StringBuilder();
    Catalog theCat = theDoc.ObjectSoup.Catalog;
    Atom theForm = theCat.Resolve(Atom.GetItem(theCat.Atom, "AcroForm"));
    Atom theXfa = theCat.Resolve(Atom.GetItem(theForm, "XFA"));
    if (theXfa is ArrayAtom)
    {
        ArrayAtom a = (ArrayAtom)theXfa;
        int n = a.Count;
        Atom dataSetAtom = null;
        for (int i = 0; i < n; i++)
        {
            Atom theItem = a[i];
```

```
                if (theItem is StringAtom)
                {
                    if ( Atom.GetText(theItem) == "datasets")
                    {
                        dataSetAtom = a[i + 1];
                        break;
                    }
                }
        }
        if (dataSetAtom != null)
        {
            IndirectObject theStm =
((RefAtom)dataSetAtom).Resolve(theDoc.ObjectSoup);
            CCRXmlDocument ccrXmlDocument = null;
            if (theStm is StreamObject)
            {
                StreamObject s = (StreamObject)theStm;
                s.Decompress ();
                XmlDocument xmlDocument = new XmlDocument();
                XmlNamespaceManager nsmgr = new
XmlNamespaceManager(xmlDocument.NameTable);
                nsmgr.AddNamespace("xfa", "http://www.xfa.org/schema/xfa-
data/1.0/");
                nsmgr.AddNamespace("dd", " http://ns.adobe.com/data-description/");
                nsmgr.AddNamespace("ccr", "urn:astm-org:CCR");
                xmlDocument.Load(new StringReader("<?xml version=\"1.0\"
encoding=\"UTF-8\"?>" + s.GetText()));
                XmlNode dataNode =
xmlDocument.SelectSingleNode("//xfa:datasets/xfa:data/ccr:ContinuityOfCareRecord",
nsmgr);
                XmlDocument ccrXmlDocument = new XmlDocument();
                ccrXmlDocument.AppendChild(ccrXmlDocument.ImportNode(dataNode,
true));
                return ccrXmlDocument;
            }
        }
    }
    return null;
}
```

INPUTTING THE XML DATA STREAM WITH XDP ON A SERVER

This section provides a quick overview on serving XML Data as an XDP, and referencing a PDF from that XDP uses to display a form or document merged with the XML data. This is useful for server side processors which are unable to manipulate PDF directly, but can manipulate XML. This section also provides sample code for doing this work using an available commercial PDF vendor's PDF Library. Full file format details are available in the *PDF Reference* and *XFA Specification*.

XML Data with the XDP File

The *XFA Specification*, version 2.4, section 21, *XDP Specification*, describes the XDP (XML Data Package) file format and syntax. XDP provides a way to package XFA components and other content in an XML-based container file. The XFA components can include an XFA document template, a PDF document, XML form data, configuration information, and other XFA or custom components.

The XDP format provides an alternative way to express a PDF document, where the outer package is in XML syntax instead of PDF syntax. Elements of the PDF are usually copied into packages in a corresponding XDP. The packages in the XDP should take precedence over the corresponding elements in the PDF. For example, if a dataset exists in an XDP, it will take precedence over a dataset in the corresponding PDF. An XDP can be thought of as a PDF

turned "inside-out", since internal components of the PDF have been exposed in XML syntax in the XDP.

Table 8 uses the sample XDP provided in the XFA Specification to illustrate XDP syntax.

Table 8: XML data as XDP

```
<xdp:xdp xmlns:xdp=http://ns.adobe.com/xdp/ uuid="…" timeStamp="1994-11-05T13:15:30Z">
<xfa:datasets xmlns:xfa="http://www.xfa.org/schema/xfa-data/1.0/">
<xfa:data>
<book>
<ISBN>15536455</ISBN>
<title>Introduction to XML</title>
<author>
<firstname>Charles</firstname>
<lastname>Porter</lastname>
</author>
</book>
</xfa:data>
</xfa:datasets>
<pdf xmlns="http://ns.adobe.com/xdp/pdf/">
<document>
<chunk> JVBERi0xLjMKJeTjz9IKNSAwIG9iago8PC9MZW5ndGQo+PgpzdHJlYW0KeJyIWEtv3DYQvutX8FFKgPZj...
ZQo+PgpzdHJlYW0KeJyIWEtv3DYQvutX8FFKgPZj...
Z/iUBGstoTDg9cfVfPPgcPjJDxUnDH7wt3GCtPv...
</chunk>
</document>
</pdf>
</xdp:xdp>
```

In the above sample, the XDP is the wrapping file. That XDP contains an XFA datasets packet (in the <xfa:datasets> tag) which will override any datasets packet contained in the PDF. The XDP also contains a PDF packet (in the <pdf> tag). The sample above contains the actual PDF under this packet (it is base64 encoded under the pdf.document.chunk element). Alternately, to avoid embedding the PDF inside the XDP, the PDF may be the external target of an href link, as in the example in Table 9.

Table 9: XDP code with PDF as an external target

```
<xdp:xdp xmlns:xdp=http://ns.adobe.com/xdp/ uuid="…" timeStamp="1994-11-05T13:15:30Z">
  <xfa:datasets xmlns:xfa="http://www.xfa.org/schema/xfa-data/1.0/">
    <xfa:data>
      <book>
        <ISBN>15536455</ISBN>
          <title>Introduction to XML</title>
          <author>
            <firstname>Charles</firstname>
            <lastname>Porter</lastname>
          </author>
      </book>
    </xfa:data>
  </xfa:datasets>
  <pdf href="pathname/filename.pdf" xmlns="http://ns.adobe.com/xdp/pdf/" />
</xdp:xdp>
```

In practice, when a server supplies the above code, a compliant PDF reader, such as Adobe Reader, will recognize the XDP, download the PDF referenced by the href, and merge the data specified in the XDP dataset into the PDF. For those familiar with the FDF data format, this works similarly to the merging of an FDF with PDF. This can be advantageous in some

scenarios since not all server processors are capable of directly merging or manipulating PDF and XFA data, though it does not provide the full functionality of XFA and PDF native tools.

XML Data with the XDP File Sample Code

Table 10is a sample C# code snippet that creates an XDP file with the above structure. It uses a C# .NET user control which can generate an XDP file for a given CCR (represented in a .NET XmlDocument class) from an XFA PDF template stored on a server: This utilizes the ABCPDF.NET library (v6).

Table 10: Sample XML data with the XDP file

```
protected void GenerateXDP(XmlDocument ccrXmlDocument,string serverPDFUrl)
{
    StringBuilder responseString = new StringBuilder();
    Response.ContentType = "application/vnd.adobe.xdp+xml";
    responseString.Append( "<?xml version='1.0' encoding='UTF-8'?>");
    responseString.Append("<?xfa generator='AdobeDesigner_V7.0' APIVersion='
2.2.4333.0'?>");
    responseString.Append("<xdp:xdp xmlns:xdp='http://ns.adobe.com/xdp/'>");
    responseString.Append ("<xfa:datasets xmlns:xfa='http://www.xfa.org/schema/xfa-
data/1.0/'>");
    responseString.Append("<xfa:data xfa:dataNode='dataGroup'>");
    responseString.Append(ccrXmlDocument.DocumentElement.OuterXml);
    responseString.Append("</xfa:data>");
    responseString.Append("</xfa:datasets>");
    responseString.Append(string.Format("<pdf href='{0}' xmlns='
http://ns.adobe.com/xdp/pdf/' />",
        serverPDFUrl);
    responseString.Append("<xfdf xmlns=' http://ns.adobe.com/xfdf/'
xml:space='preserve'>");
    responseString.Append("<annots/>");
    responseString.Append("</xfdf>");
    responseString.Append("</xdp:xdp>");
    Response.Write(responseString);
    Response.Flush();
    Response.End();
}
```

For more information on this topic, view the discussion on Adobe Forums found at: http://www.adobeforums.com/cgi-bin/webx?14@@.3bb9d6c1/0.

FORMS

The forms found in Figure 7 and Figure 8 illustrate the sample form "Octavia_CCR_PDFXMLForm.pdf".

Patient History of   CCR ID:

Identification Information		
ID	Type	
First	Middle	Last
Gender	DOB	
Source Name:	Source Role:	

Address Information	
Line 1	
Line 2	Type
City	State
Country	Postal Code

Telephone Information
Number	Type

Email Information
Email	Type

Social History Information		
Type	Value	Source

Alert Information						
Date Type	Date	Type	Agent	Reaction	Severity	Source

CCR Date: Page 1 of 2

Figure 7: Octavia_CCR_PDFXMLForm.pdf (page 1)

Figure 8: Octavia_CCR_PDFXMLForm.pdf (page2)

EMBEDDED DICOM PDF HEALTHCARE SAMPLES

These samples were created using the following:

Tools:

- Adobe Acrobat 8.1

- Adobe LiveCycle Designer 8.05

- DesAcc Health Data Explorer 1.0

Other:

- Radiological Society of North America (RSNA), *DICOM Test Dataset* <ftp://ftp.erl.wustl.edu/pub/dicom/images/version3/RSNA96/>

References:

- *Health Data Explorer User Guide* <ftp://ftp.desacc.com/hde1_manual.pdf>

- *NEMA PS 3-2007, Digital Imaging and Communications in Medicine (DICOM)*, version 3.0 [18 parts] <http://medical.nema.org/dicom/2007/>

- *Digital Imaging and Communications in Medicine (DICOM)*, *version 3.0, Supplement 104: DICOM Encapsulation of PDF Documents* <ftp://medical.nema.org/medical/dicom/final/sup104_ft.pdf>

Samples:

The **IG-Kit\PDF-DICOM_samples** folder contains the source files that Health Data Explorer utilizes in creating the file **2x3.pdf**.

Table 11: Embedded DICOM sample files

File	Description
2x3_template.xdp	XML Data Package (XDP) containing XFA template syntax for this form
Image01.dcm through Image12.dcm	Source DICOM images selected by the user within the Health Data Explorer Local Storage tab for building embedded DICOM PDF file
2x3_template.xml	The intermediate XML data created by Health Data Explorer as it converts the DICOM files to a PDF utilizing the XFA template

EMBEDDED DICOM PDF GENERATION GUIDELINES

Assumptions

The following section assumes the user is working with Acrobat 8.1 and the DesAcc Health Data Explorer 1.0 plug-in. The dataset used has been loaded from a sample DICOM CD, but could also have been acquired directly from a remote DICOM device through a DICOM Query/Retrieve operation or by the remote DICOM device initiating a DICOM Send operation to Acrobat 8.1 running the plug-in.

Background

By creating a mapping between the individual DICOM elements that make up a DICOM file and elements within a user definable XFA form, it is possible to trans-code the header items within a DICOM file into an XFA form element, allowing for PDF forms to be auto-populated based on the user's XFA form definition. The below figure shows an example of the elements and hierarchical tree structure that makes up a standard DICOM file.

Figure 9: Hierarchical Display of a Typical DICOM File

DICOM elements themselves are rigidly structured by the requirement of the DICOM 3.0 Standard, and have a specified encoding structure and multiplicity that allows for a well formed XML Schema to be designed and utilized within Adobe LiveCycle Designer 8.0. This allows the user to define any template form within Designer where DICOM elements will map directly from the original files into the XFA Form. The actual mapping of XFA elements to the DICOM element schema can be seen below within LiveCycle Designer's Data View tab.
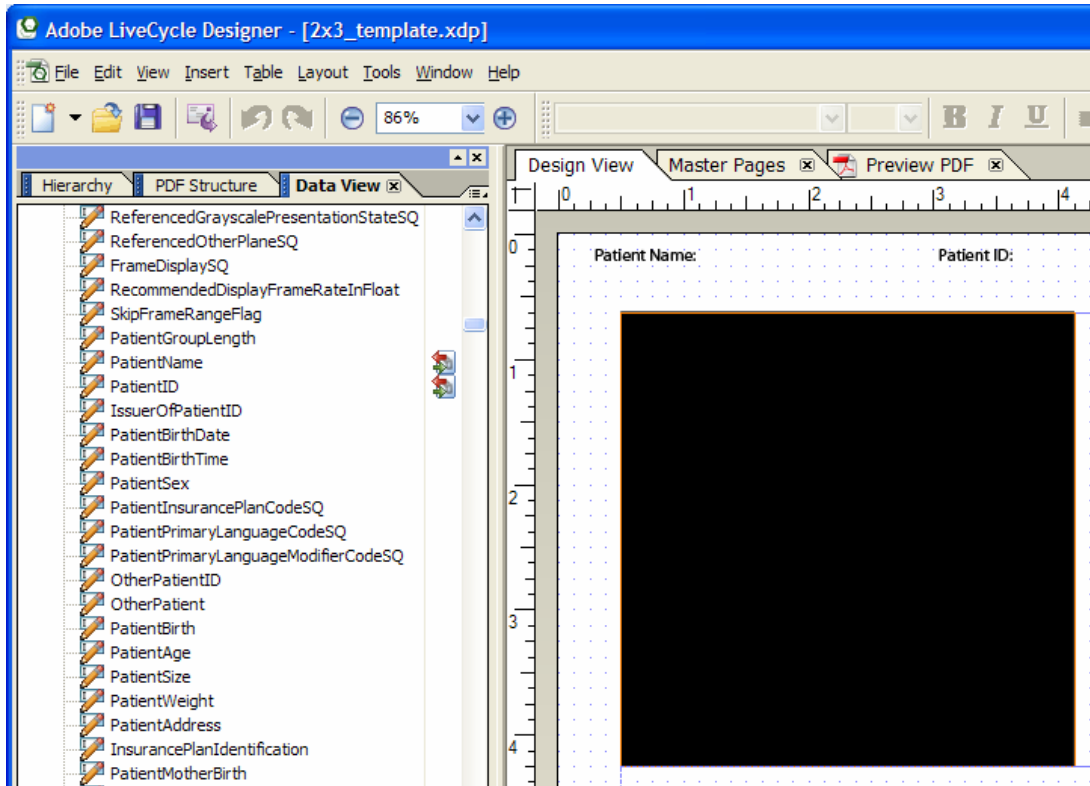
Figure 10: DICOM Schema Displayed within Adobe LiveCycle Designer 8.0

The DICOM elements, which encompass patient demographics, study information, medical device identifiers and the actual layout of the image data, allow spring loaded JavaScript logic within an XFA form to be utilized for rendering sub-forms specific to information found within the DICOM file, such as adding a PDF form section related specifically to the Modality of the data (Modality = "US" above, meaning Ultrasound data), or specific to patient age, for example. As the header also distinctly identifies how to decode the image or images stored within the DICOM file, this information can be used to generate JPEG representations of the data which can be loaded into the XFA form through base64 encoding of the data within an XML source file. An example of the actual DICOM elements that will be mapped into the PDF Form are shown below within the Hierarchy tab of Adobe LiveCycle Designer.
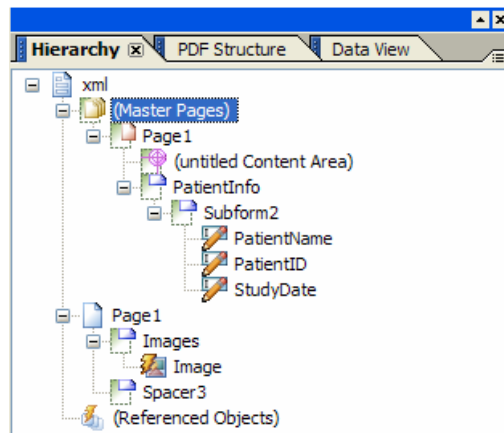

Figure 11: XFA Form Page Hierarchy with Image Field and Mapped Element Fields

Given this need for being able to access, store, and transparently map between DICOM space and PDF space, Health Data Explorer was developed for directly allowing Adobe Acrobat to integrate DICOM functionality into PDF workflow.

Operation

The majority of medical imaging data collected on today's medical scanners is stored and shared using the Digital Imaging and Communications in Medicine (DICOM) 3.0 protocol. DICOM corresponds to both a file format for archiving the image data as well as a transfer protocol for network access to DICOM assets. The Health Data Explorer plug-in to Acrobat 8 has been developed for both originating data, in the form of DICOM Encapsulated PDF Files (DICOM Supplement 104), which can encapsulate any PDF file within a DICOM compliant wrapper for transmission and storage, as well as for accessing DICOM image files through DICOM network protocols.

As a single medical imaging study can contain thousands of images, such as those generated by current spiral CT technology, a local database of DICOM files is required for storing data transmitted from medical devices, and is accessed directly from the Acrobat "File" menu, as illustrated in Figure 12.
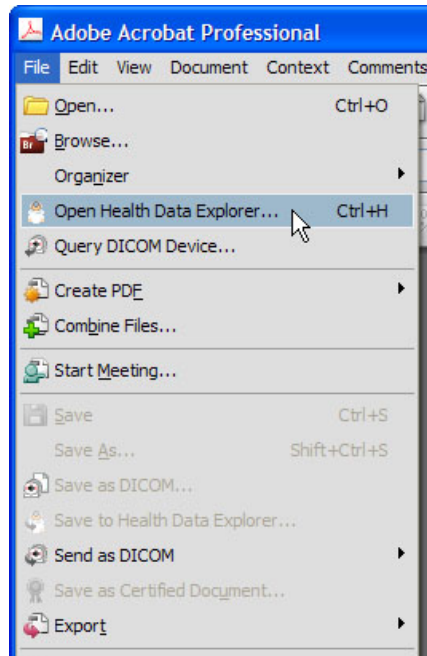

Figure 12: DICOM Menu Items Added to Adobe Acrobat

In addition to accessing locally stored data, the ability to access remote DICOM devices (File > Query DICOM Device…) and retrieve data from the remote device, such as an MR scanner or Picture Archiving and Communications System (PACS) allows Adobe Acrobat to operate as a full DICOM node within a hospital's DICOM network.

From the "Local Storage" tab of Health Data Explorer, the DICOM image data can be selected based on a user's search criteria and then used for creating PDF files that contain both JPEG renditions of the source DICOM image data as well as embedded copies of the original source data, as shown in Figure 13.
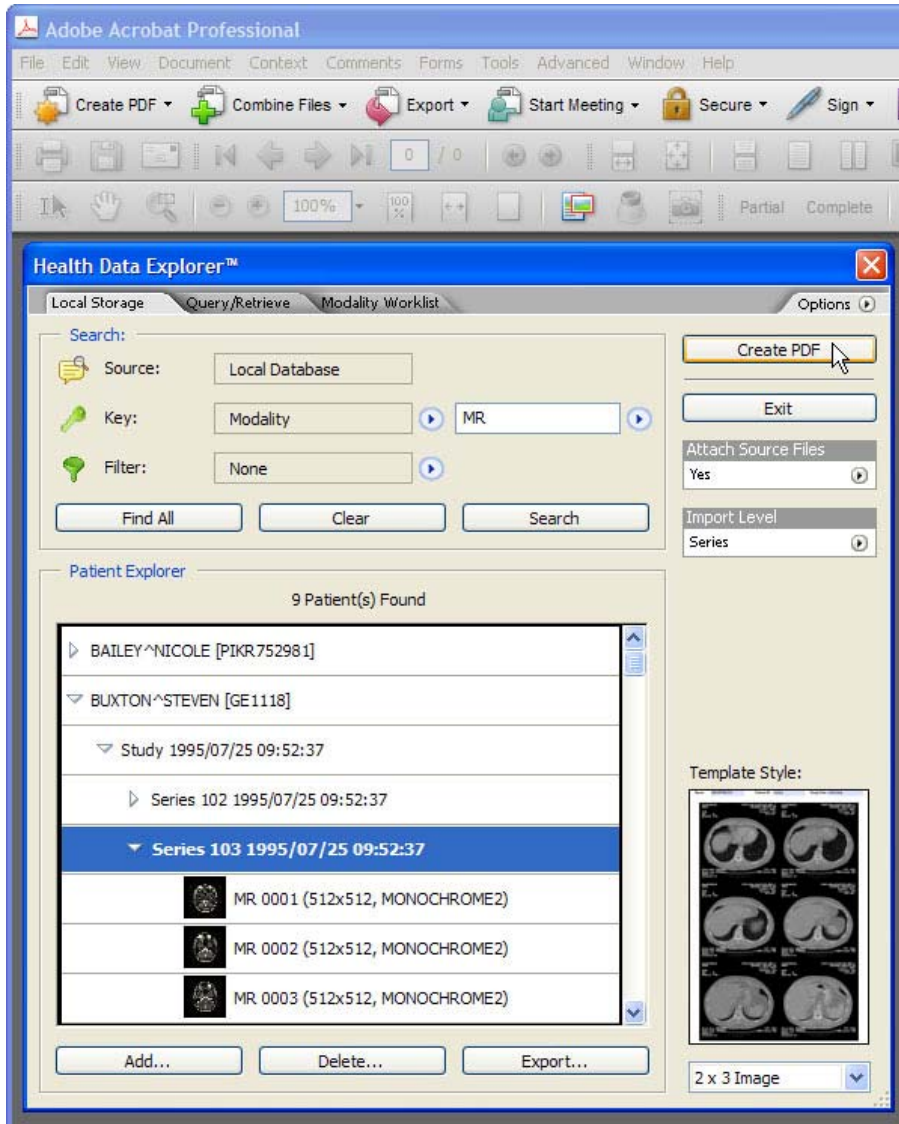
Figure 13: Health Data Explorer Image File Selection

The DICOM Standard sets out a hierarchical structure of Patient / Study / Series / Image for managing patient data. This is shown within the Health Data Explorer, and allows PDF files to be created based on a user defined selection of individual series or studies. A set of user expandable XDP templates is used for guiding the mapping between DICOM elements and PDF form objects. These forms can load text from the DICOM header, DICOM images, or a combination of the two.

In addition, the user can specify whether the created PDF will contain the source DICOM image files as PDF attachments, allowing the original data as a faithful archive to accompany the PDF file (see Figure 14). This DICOM data can then be transmitted at a later time or exported locally where diagnostic reading of the images can be performed.

PDF files created by the Health Data Explorer plug-in can utilize the dynamic features of XFA forms to automatically create additional pages as necessary based on the selected user data.
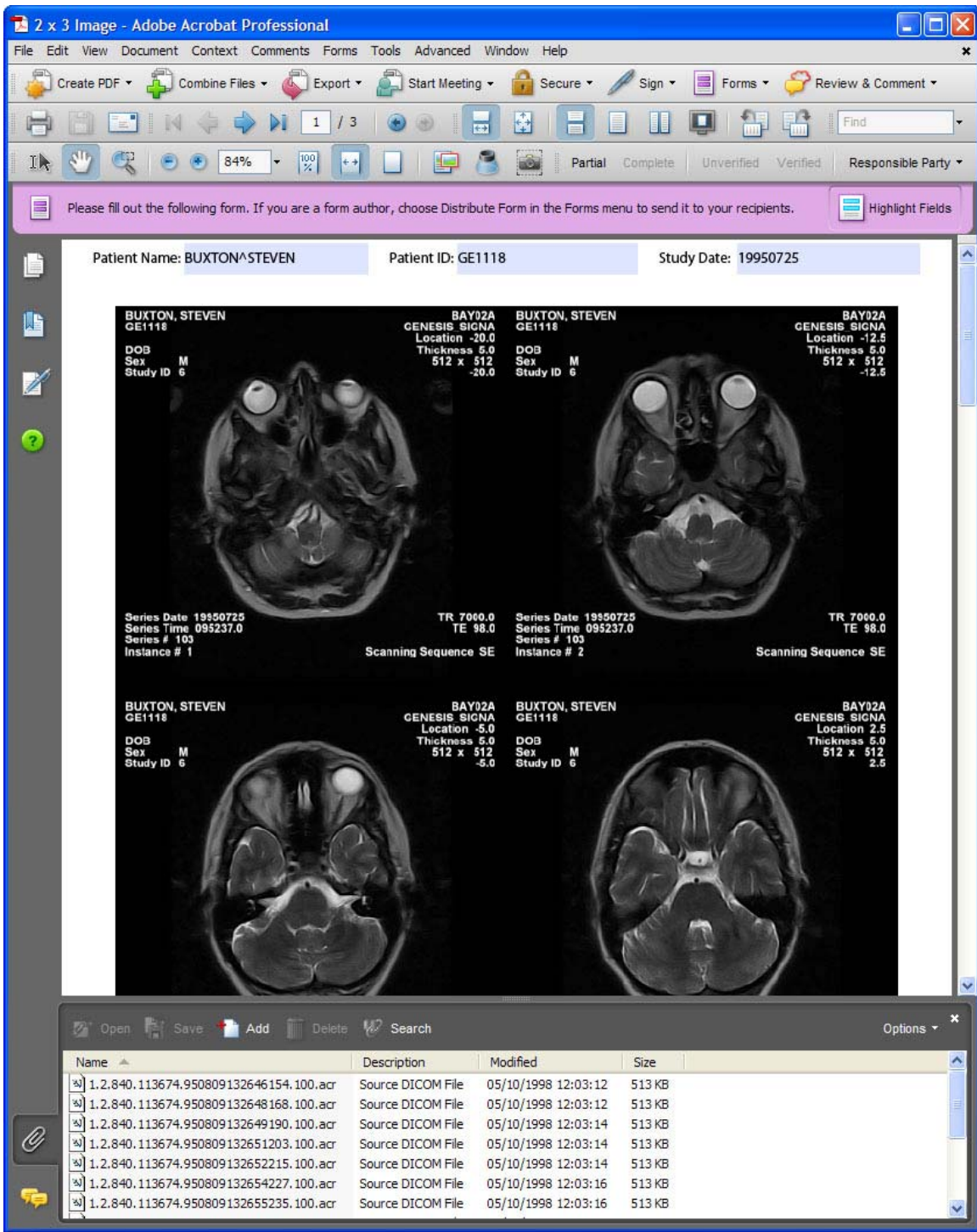
Figure 14: Health Data Explorer DICOM Image File Attachment

Once the data has been mapped into the PDF, the full functionality of Adobe Acrobat can be used for assigning password protection, digital signatures for ensuring integrity of data, and even group level access privileges through Adobe LiveCycle Rights Management ES, which can define access privileges to PDF document workflow. With LiveCycle Rights Management, all access to the PDF file can be logged, as specified within the Health Insurance Portability

and Accountability Act (HIPAA), including the ability to retract access privileges to users having physical copies of the rights enabled PDF files.

APPLYING CLINICAL CONTEXT TO PDF FILES

These samples were created using the following:

Tools

- Adobe Acrobat 8.1

- DesAcc Health Data Explorer 1.0

References

- *Digital Imaging and Communications in Medicine (DICOM), version 3.0, Supplement 104, DICOM Encapsulation of PDF Documents*
  <ftp://medical.nema.org/medical/dicom/final/sup104_ft.pdf>

- ANSI/HL7 V2.3.1, *Health Level Seven Standard Version 2.3.1*
  <https://www.hl7.org/library/standards_non1.htm#HL7%20Version%202.3.1>

- Adobe Extensible Metadata Platform (XMP)
  <http://partners.adobe.com/public/developer/xmp/topic.html>

BACKGROUND

Medical devices operating within the healthcare environment utilize standards for ensuring that patient specific data such as lab results and medical images are properly labeled and identified, i.e. the data has a "clinical context." This context specifies patient demographics, collection conditions, and unique reference identifiers. For radiological devices, this information is generated by the hospital or clinic's Radiology Information System (RIS) using the DICOM standard. For pathology data, clinical tests, and non-image related data, the context is taken from the Hospital Information System (HIS), which uses the HL7 standard for communication. As a large amount of patient-centric documents, such as consent forms and notes taken by doctors, nurses, surgeons, and technicians is still paper based, moving this data into digital form means that a clinical context must be applied to the corresponding electronic files.

OPERATION

The Health Data Explorer plug-in for Adobe Acrobat 8 allows Acrobat to interact directly with the information systems that provide clinical context for patient related data collection. By directly accessing the systems that are the backbone of clinical care, any PDF document can become part of the health information workflow within a hospital environment. As both the DICOM standard (Supplement 104) and HL7 standard (version 2.3.1 and greater) have added full support for encapsulating PDF files within their information systems, once clinical context has been applied to a PDF it can be transported in and out of the DICOM and HL7 systems within a hospital. In addition, by embedding the context information within the PDF document using Adobe's Extensible Metadata Platform (XMP) metadata technology, full searches on clinically significant information can be performed.

It is from the "Context" menu added by Health Data Explorer (see Figure 15) that a user can either directly communicate with a hospital system's demographic server or manually enter the context through a user driven wizard that allows for the collection of the required clinically significant information.
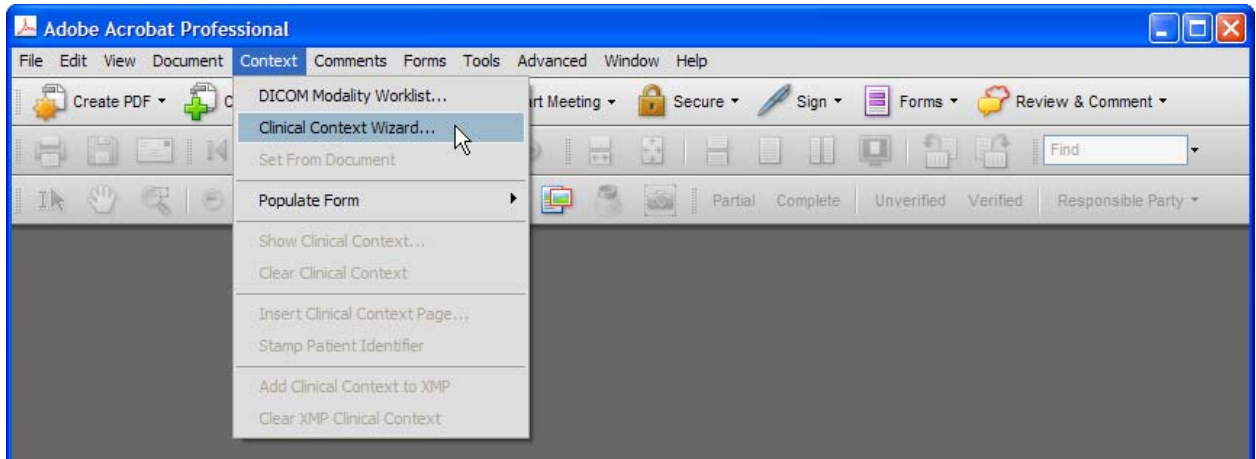
Figure 15: Health Data Explorer "Context" Menu

If the Clinical Context Wizard is used to enter the patient demographics, a number of wizard screens prompt the user for the appropriate input (see Figure 16 and Figure 17).
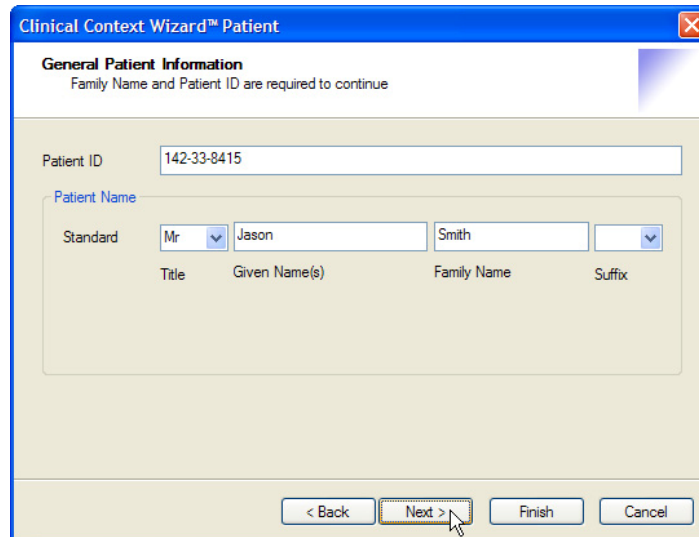

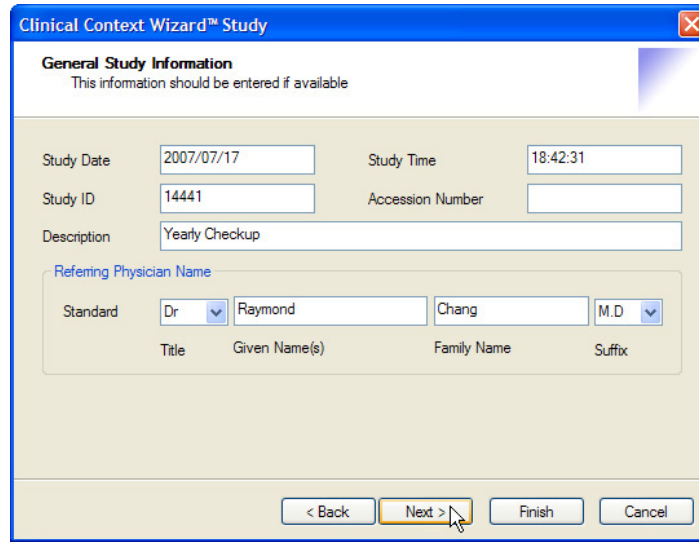**Figure 16: Health Data Explorer Clinical Context Wizard Patient Information Screen**

Figure 17: Health Data Explorer Clinical Context Wizard General Study Information Screen

While data can be entered manually, for environments with RIS or HIS systems it is possible to query the demographics server directly. By setting the context directly from the RIS (through a DICOM Modality Worklist query as shown in Figure 18), Acrobat is able to tie directly into the enterprise servers that also feed MR, CT, Ultrasound, and other health imaging systems.
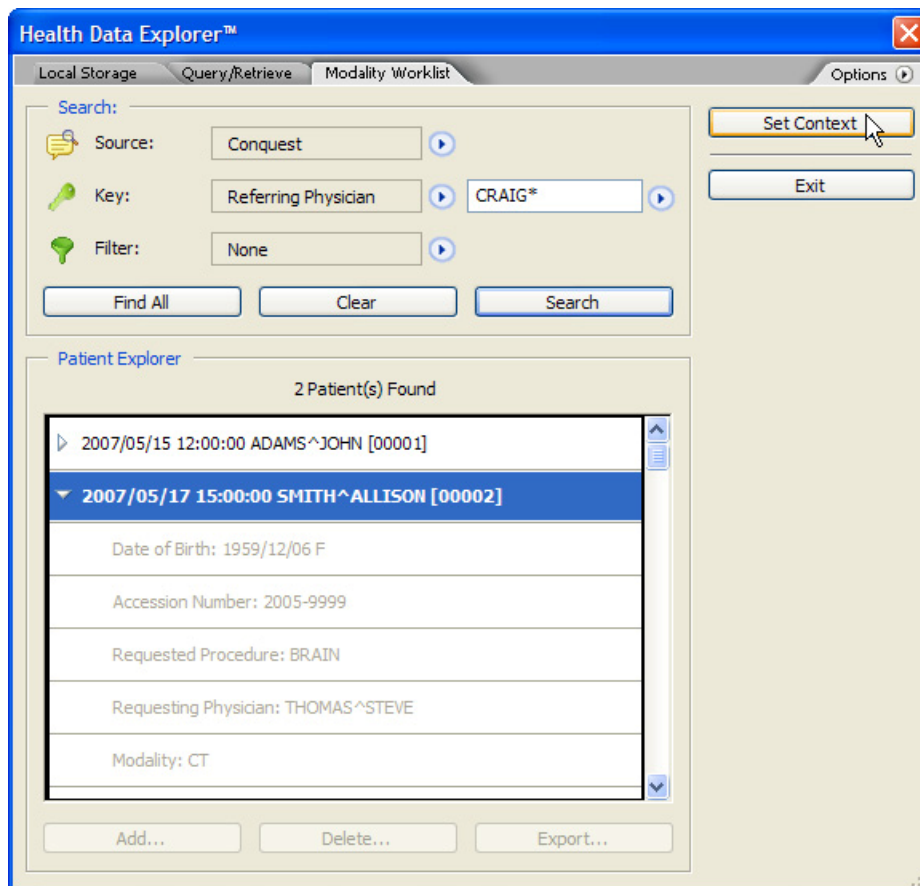
Figure 18: Health Data Explorer DICOM Modality Worksheet

When the clinical context information has been specified, a PDF file can be "stamped" with an identifying patient header (see Figure 19 and Figure 20) and have the context added to its XMP metadata.
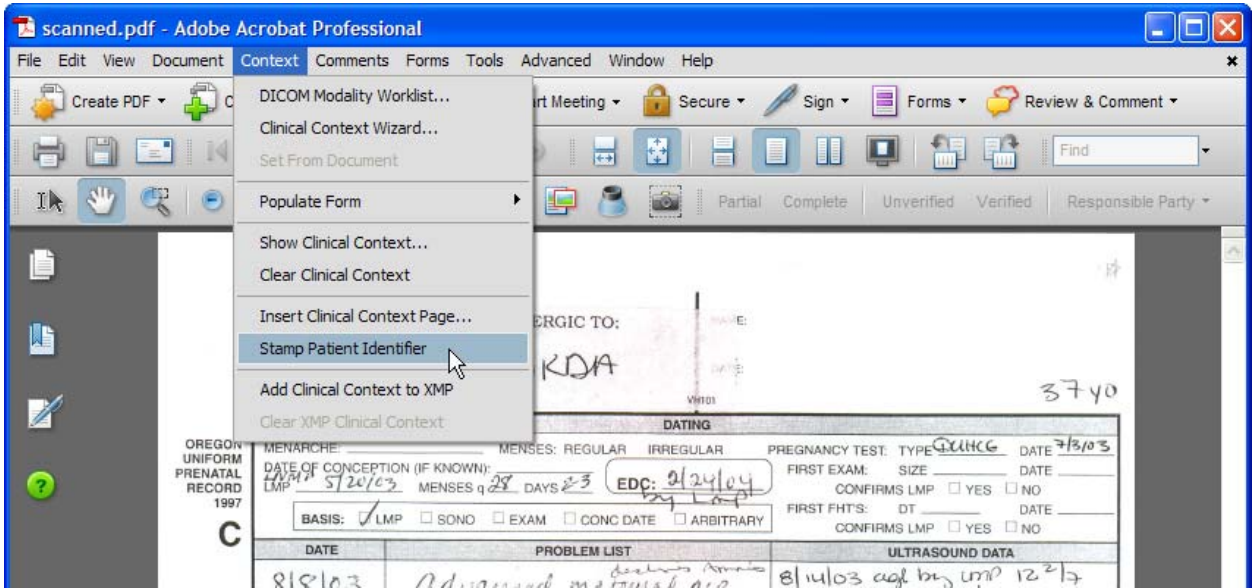

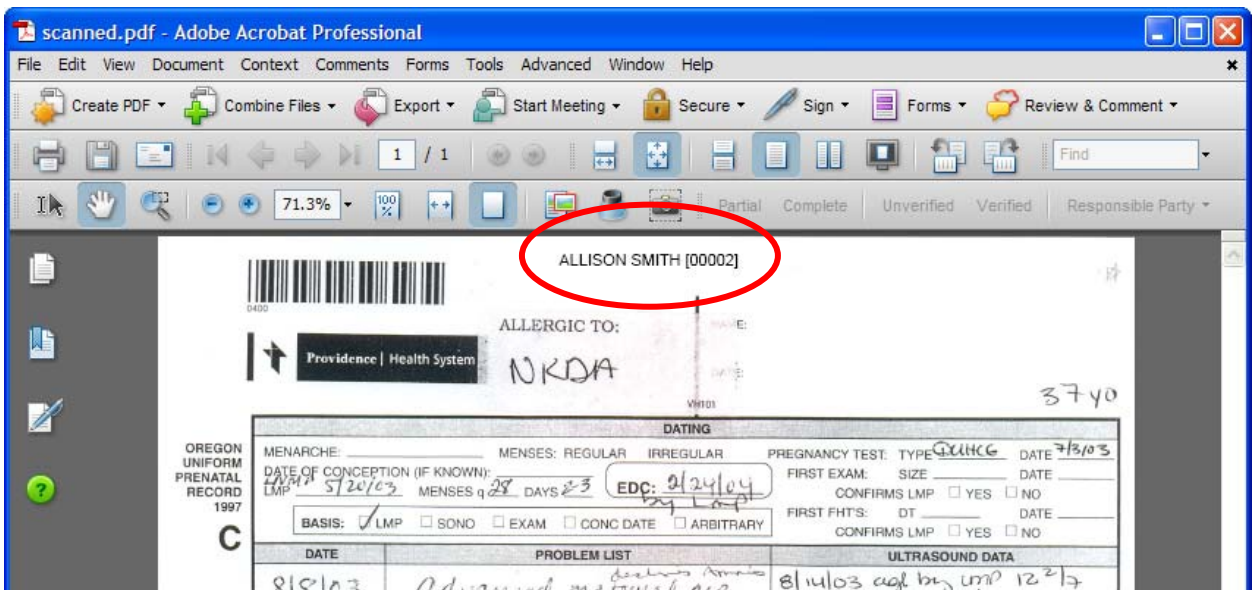**Figure 19: Health Data Explorer Patient Identifier Stamp Selection**

Figure 20: Health Data Explorer patient Identifier Stamp Displayed on Document

The clinical context can also be used to fill out pre-defined forms, such as patient consent forms, so that identifying patient information does not need to be manually entered for each form.

Once a PDF document has a clinical context, it can also be transmitted and stored within the hospital environment using the DICOM standard. This is done either through saving the document to a "DICOM Encapsulated PDF" file or transmitting it to a remote clinical system using the DICOM network protocol, which is done transparently by the Health Data Explorer plug-in (see Figure 21). Provisions are also available for querying such DICOM Encapsulated PDF files from a remote DICOM device, extracting the original PDF and displaying it within Acrobat, with the full security and digital signatures that are available with Acrobat 8.
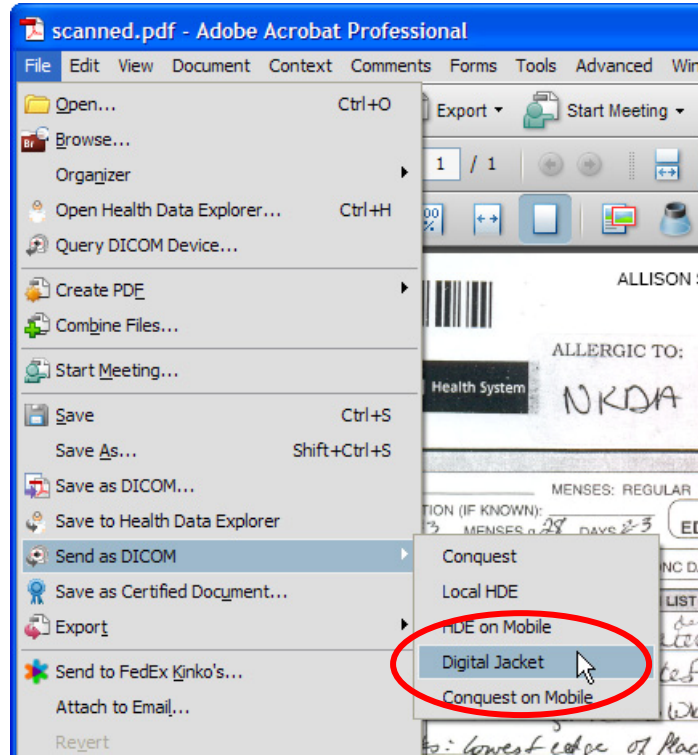

Figure 21: Health Data Explorer DICOM File Transmission

CONCLUSION

Without context, one piece of paper cannot be distinguished from another. The same can be said for electronic documents. Without context they are just icons on a computer's desktop. By providing Acrobat with the ability for a PDF file to state, "I belong to patient John Smith and this is my Patient ID," the PDF file is enabled as a participant in a hospital's electronic document workflow. By allowing Acrobat to participate in a hospital's electronic data workflow, any paper document, electronic form, or scanned document can be provided with clinical context, enhancing the patient experience.

BIBLIOGRAPHY

Adobe XML Form Object Model Reference (for Adobe LiveCycle Designer, version 7.1). Adobe Systems Incorporated Inc., 2005.
<http://partners.adobe.com/public/developer/en/xml/Adobe_XML_Form_Object_Model_Reference.pdf

AIIM BP-01-200X, *Portable Document Format in Healthcare A Best Practices Guide.*

Americans With Disabilities Act (42 U.S.C. 12101 et seq.), U.S. Public Law 101-336.

ANSI/HL7 V2.3.1, *Health Level Seven Standard Version 2.3.1*
<https://www.hl7.org/library/standards_non1.htm#HL7%20Version%202.3.1>

ANSI/NISO Z39.85-2007, *Dublin Core Metadata Element Set*. National Information Standards Organization, May 2007.
<http://www.niso.org/standards/standard_detail.cfm?std_id=725>

ASTM E2369-05, *Standard Specification for Continuity of Care Record (CCR).* ASTM International, 2005.

Digital Imaging and Communications in Medicine (DICOM), version 3.0, Supplement 104: DICOM Encapsulation of PDF Documents
<ftp://medical.nema.org/medical/dicom/final/sup104_ft.pdf >

ECMA-363, Universal 3D File Format, 4th edition. Ecma International, June 2007.
<http://www.ecma-international.org/publications/standards/Ecma-363.htm>

Health Data Explorer User Guide. DesAcc, Inc.
<ftp://ftp.desacc.com/hde1_manual.pdf>

Health Insurance Portability and Accountability Act of 1996, U.S. Public Law 104-191, August 21, 1996.

ISO/WD 19005-2, *Document management – Electronic document file format for long-term preservation, Part 2: Application of PDF 1.6.* International Organization for Standardization, committee draft; not yet published.

NEMA PS 3-2007, Digital Imaging and Communications in Medicine (DICOM), version 3.0 [18 parts]
<http://medical.nema.org/dicom/2007/>

PDF Reference: Adobe Portable Document Format, Version 1.6, 4th Edition. Adobe Systems Inc., December 2004 (ISBN: 0-321-30474-8).
Latest version available from: <http://www.adobe.com/devnet/acrobat/pdfs/pdf_reference.pdf>

PDF Reference: Adobe Portable Document Format, Version 1.7, 6th Edition. Adobe Systems Incorporated Inc., November 2006.
Latest version available from: <http://www.adobe.com/devnet/acrobat/pdfs/pdf_reference.pdf>

Radiological Society of North America (RSNA), *DICOM Test Dataset*
<ftp://ftp.erl.wustl.edu/pub/dicom/images/version3/RSNA96/>

*Section 508 of the Rehabilitation Ac*t (29 U.S.C. 794d), P.L. 105-220.
<http://www.section508.gov/index.cfm?FuseAction=Content&ID=14>

W3C Recommendation, *Web Content Accessibility Guidelines 1.0*. World Wide Web Consortium,
May 5, 1999. <http://www.w3.org/TR/WCAG10/>

W3C Recommendation, *RDF/XML Syntax Specification*. World Wide Web Consortium, February 10,
2004. <http://www.w3.org/TR/rdf-syntax-grammar/>

XML Forms Architecture (XFA) Specification, Version 2.4. Adobe Systems Incorporated Inc.,
September 11, 2006. <http://partners.adobe.com/public/developer/en/xml/xfa_spec_2_4.pdf>

XMP Specification. Adobe Systems Incorporated Inc., September 2005.
<http://partners.adobe.com/public/developer/en/xmp/sdk/XMPspecification.pdf>