Meeting Minutes

PWG MFD Semantic Model Face-to-Face
September 28, 2007

Attendees:

Ron Bergman	Ricoh
Nancy Chen	Oki Data
Lee Farrell	Canon
Grant Gilmour	366 Software
Harry Lewis	InfoPrint
Glen Petrie	Epson
Jerry Thrasher	Lexmark
Bill Wagner	TIC
David Whitehead	Lexmark
Peter Zehler	Xerox

Meeting Agenda Today:

1. Validate resolutions and resolve open issues from September 26 Meeting.
2. Do we want signed document or other security attributes in document content? The same applies to template.
3. Approaches to support capabilities in XML schema
4. Flesh out operations and data elements required for Scan Service by walking through use cases.

Validate Resolutions and Resolve Issues Raised from September 26 Meeting –

1. Action items from editorial changes:
   - Glen Petrie to send Nancy the remaining editorial changes.
   - Peter Zehler to supply the definition of Template Repository.
2. Action items from the detailed state transition diagram:
   - Peter Zehler to rework the diagram.
3. Issue on how should Scan Service handle authentication to Document Repository when storing client's document data?:
   - Resolution was: The user needs to ensure that Document Repository is administered to allow Scan Service to write the user's data and restrict read access as appropriate, using the Scan Service's own identity, not to assume the user's identity. The user must specify the directory (folder) as the destination of the scan document. Then the Scan Service provides username as part of the filename to be written into the directory specified by the user.
   - Issue: How is the Scan Client informed of the location (filename) of his scan document?
     - Discussion: We have already decided that URL must be used for specifying the destination of scan document. If a folder URL is specified, the Scan Service will need to generate the actual file URL of the scan document and notify the scan client the file location (URL) of the scan document. This requires an element for

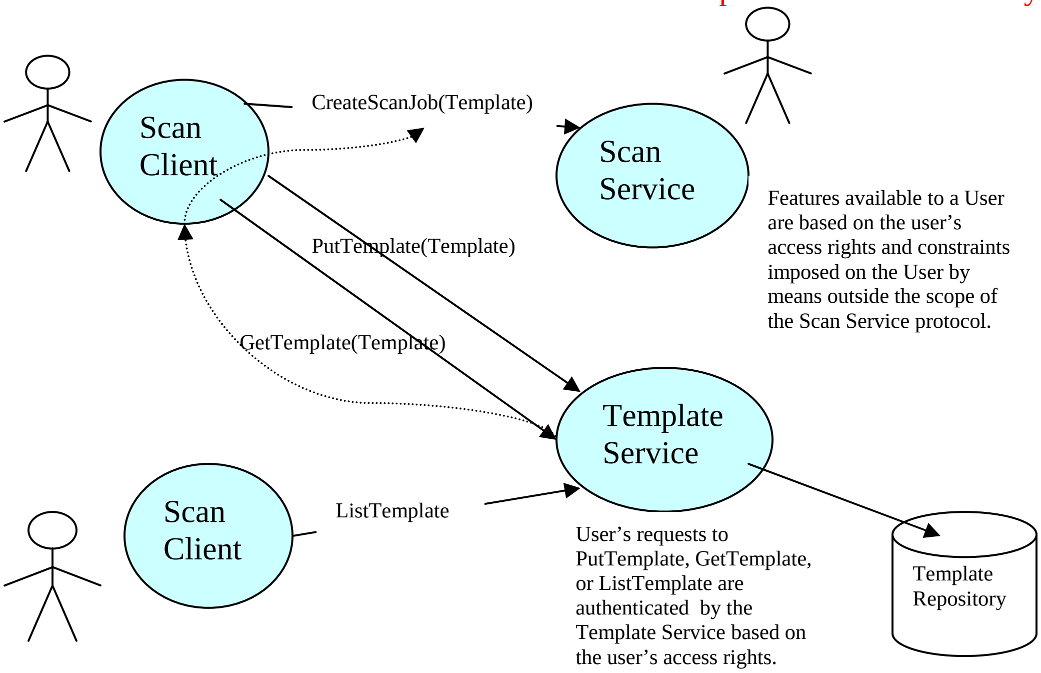
Scan Service to hold the file name of the scan document in order to notify the scan client. If the user selected “Multi-Document Mode”, then there will be a sequence of filenames been recorded and returned to the user. In this scenario, the user needs to specify the scan destination as a folder, not a file. When the job is completed, the Scan Service will notify the client. The client can then use “GetJobElement” to request the Scan Service to return all information of the completed job including the destination URL plus one or more filenames of the scan document depending on “Single Document” or “Multi-Document” mode selected by the user. The implementation for the Scan Service may choose to not maintain the records of scan document file locations and dynamically generate them when being asked by the client. The “Folder as scan destination URL” approach will preclude user’s ability to choose his own document name. To allow user to specify either a folder or his own document name for the scan destination, the XML schema for the destination element can be modified to a choice of folder URL or file URL.

- Consensus: The Scan Service XML schema will be modified to use “choice” of folder URL or file URL for the scan destination element. At the job completion, the Scan Service will notify Scan Client the end-of-job event. The Scan Client can then use “GetJobElement” request to obtain all job related information including the document file URL or the document folder URL with all document file names supplied by the Scan Service.
- Issue: What does the Scan Service return in response to CreateScanJob request? The Scan Service might not be able to determine the final location(s) of scan document(s) until all hardcopies have been scanned.

4. Issue: Is there a need for Restricted Use Scan Template (i.e. restrict the access and use of template stored in the Template Repository to authorized End Users)?
   - Discussion: A user could create a template to which access is restricted. A template could be created for public for unrestricted read access, or for a specific group of users for restricted read access by the users within the group, but nobody should be allowed to modify the template unless he is the owner of the template. A user who can read the template can make a copy of the template and modify the copy of course, but not the original template. Therefore Template Service must authenticate the PutTemplate(template) and GetTemplate(template) requests at the protocol level using, for example, username/password challenge or credentials required for TLS if TLS protocol is used. After a user got a template from the Template Service, presumably he can modify the template then pass it into the CreateScanJob request to the Scan Service. To enforce a site policy that does not allow users to modify the template, the Scan Service must allow the Scan Client to pass the reference to the user’s template in the CreateScanJob request. The Scan Service then uses the reference to retrieve the template from the Template Service and checks the user’s context against the ‘RequestingUserName’ of the template to determine whether the CreateScanJob request should be accepted. In this semantics, the Scan Service must always be allowed to read a template from the Template Service. However this might not be possible when the Scan Service and the Template Service reside in different domains. The alternative is the Scan Service maintains the user’s template access security context. The Scan Client always retrieves a copy of template on behalf of a

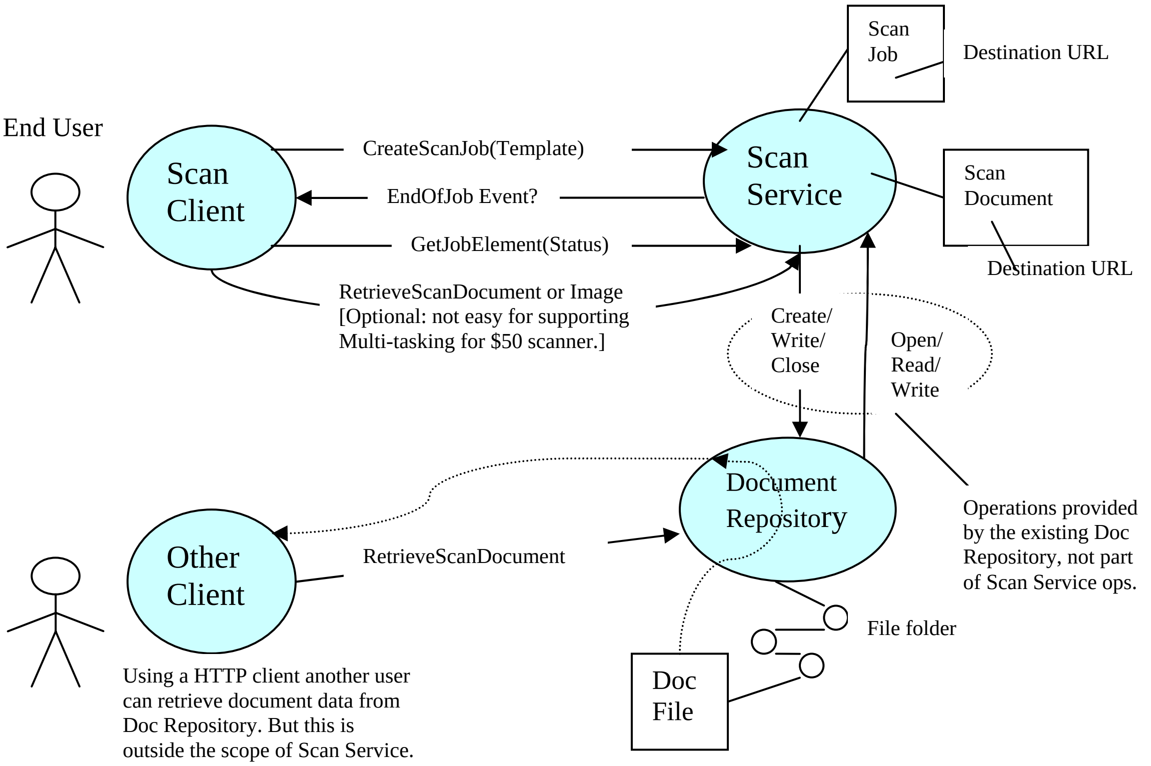
user, and passes that into CreateScanJob request. The Scan Service then uses the “RequestingUserName’ in the template and checks the contents of the template and rejects or accepts the job based on the user’s security context that is obtained from the security framework or security database of the Scan Service.

- Resolution: We do not need to add a “restricted use” flag in the template. The template must be named, must have user’s identification, and must be typed. A user will be authenticated for PutTemplate, GetTemplate, and ListTemplate by the Template Service. A user always obtains a template from the Template Service; he may modify the template before sending the CreateScanJob that includes the template. The features (i.e. processing elements) available to Scan Client may be limited based on the user name passed in the CreatScanJob request on the user’s security context which can be obtained from the security framework or the local security database of the Scan Service. The management of the feature’s policies (e.g. user based, time of day) are outside the scope of this specification.
  This eliminates the requirement for the Scan Service to retrieve the user’s template from the Template Service and the associated template access control that must be considered between the Scan Service and the Template Service for security.


Scan Client
CreateScanJob(Template)
Scan Service
Features available to a User are based on the user’s access rights and constraints imposed on the User by means outside the scope of the Scan Service protocol.
PutTemplate(Template)
GetTemplate(Template)
Template Service
Scan Client
ListTemplate
User’s requests to PutTemplate, GetTemplate, or ListTemplate are authenticated by the Template Service based on the user’s access rights.
Template Repository

5. Issue: Where should Scan Template identification information be carried?
   - Discussion: We need template name and template originating user name for authenticating a user the use of a “restricted use” template. We can use the JobName and JobRequestingUserName in the template for TemplateName and TemplateOriginatingUserName before the template becomes ticket when bound to the job. We can also define separate elements for TemplateName and TemplateOriginatingUserName.
   - Consensus: Define new elements for TemplateName and TemplateOriginatingUserName to avoid confusion that could incur later for the overloaded semantics for JobName and JobRequestingUserName.

6. Issue: What security attributes are required for the scan service protocols?
    - We agreed to the resolution from September 26 meeting: We already have the RequestingUserName that carries an unauthenticated user name. This user name can be replaced with a more trusted value if one is available. Implementations are free to use a variety of security framework for user authentication.
7. Issue: Is there a need for signed documents?



- Discussion: Signing or encryption of stored and transmitted documents will need to be supported in the Scan Service. This will require the Scan Service to indicate signing and encryption support, and elements for public key and private key, and possibly encryption key. The scan client will need to include its signing and encryption intent. The document must include the applied signing and encryption information. However if a document is encrypted in document repository, there will be issue for other client to retrieve the encrypted document because this other client does not know how the document was encrypted. Only the Scan Client and the Scan Service know how to decrypt the document. But if the Document Repository handles encryption, then the other client can decrypt the document. We believe in general the Document Repository has its existing capabilities of signing and encryption of documents that should be utilized by the Scan Service. Whenever a document is stored, the document Repository encrypts/signs the document, and decrypt/verify on document retrieval. When there is a need to sign/encrypt the document transmitted over the network, the Scan Service negotiates an existing secure transmission protocol supported by the Document Repository that signs and encrypts the document over the network. When the document is received by the Document Repository, it is automatically decrypted/verified by the secure transmission protocol. The Document Repository then signs/encrypts the document before storing the document if so is indicated in the received document. Therefore the Scan Service does not need to perform encryption or signing of documents. This is how P2600 project partitions hardcopy device security functions so far too.

- Consensus: The Scan Service should utilize the existing encryption and signing capabilities of the Document Repository for signing/encrypting the stored documents. The Scan Service only supports signing and encryption of documents being transmitted over the network between the Scan Service and the Scan Client or Document Repository. The Scan Service must support elements that indicate its signing and encryption support capability. The Scan Client must include its signing and encryption intent when requests for CreateScanJob(template). The "Must Honored" property is used to reject or do best effort support when the Scan Service can not match the Scan Client's signing/encryption intent. If a job has multi-document files, each document must contain signing/encryption information.

Unified data type for semantic elements used in Tickets, DefaultScanTicket and Capabilities –

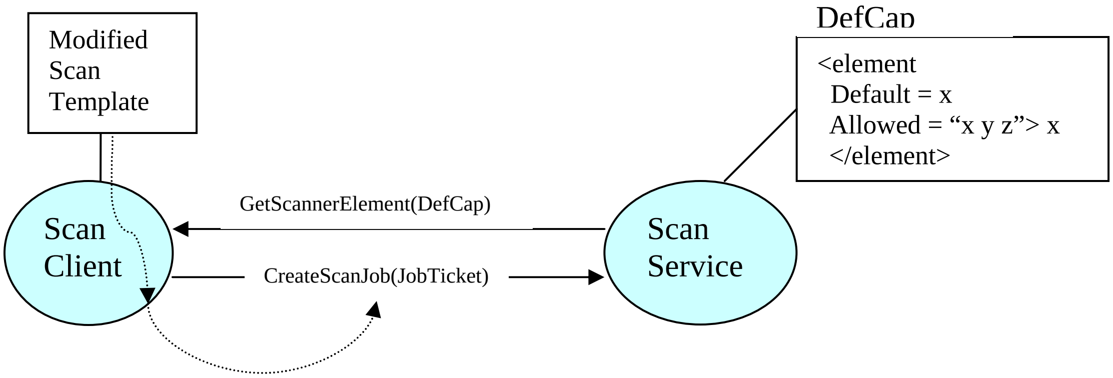
- Discussion: There are three alternative ways to represent capabilities in Scan Ticket, Default Scan Ticket in XML schema –

  1. Use xxx-supported element that has a different type than xxx element: i.e. change the syntax from NMTOKEN value used in xxx to a list of NMTOKENs for xxx-supported. This is used currently to maintain a flat name space in schema.

  2. Use decorated element with allowed values: i.e. the element has minimum and maximum values and NMTOKEN value (for current value). e.g. <MediaInfo AllowedValues = "paper polyester dry-film">paper</MediaInfo>.
  When the structure is passed as default ticket to the Scan Client, these allowed values will simply be ignored. If it's passed as "capabilities", the value will be ignored.

  3. Use local redefinition of xxx with new type: i.e. in addition to the value of xxx, the type adds a sub-element which is a sequence of allowed values, but the name of the element remains the same. The value is the default for default ticket, the internal value for ticket.

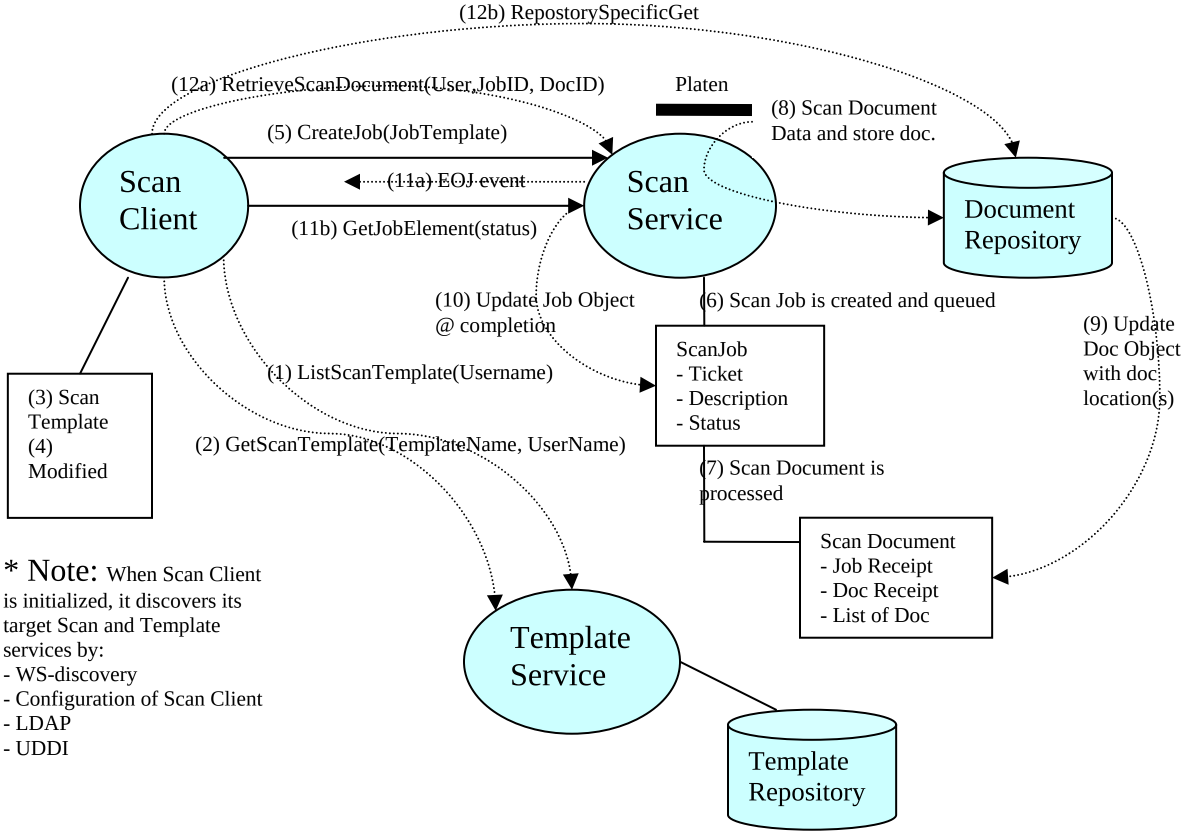
  Right now there is an error for #2 when the element has min and max attributes but no value. This still needs to be investigated. One question raised was whether we can combine the default attribute into the decorated element in #2. The value of the element is the current value. This is no problem, but if we allow the entire structure to pass around through out the entire job lifecycle, the allowed value list potentially could be quite large.



For CreateScanJob, we only want to pass the value of the element in template. However this is not possible for #2 approach. Both #1 and #3 approaches allow to pass the value only. For approach #3, template has all information in Scan Service, but ticket information itself must be bare. On the wire you will see the template is exactly the same as the ticket unless you explicitly ask to carry type info. According to Pete’s knowledge, most of XML development tools, e.g. .NET, jisaw, etc., don’t carry type information.

- Decision: Use approach #3. Approach #2 provides a lot of programming conveniences, but information that passed on the wire can be quite large.

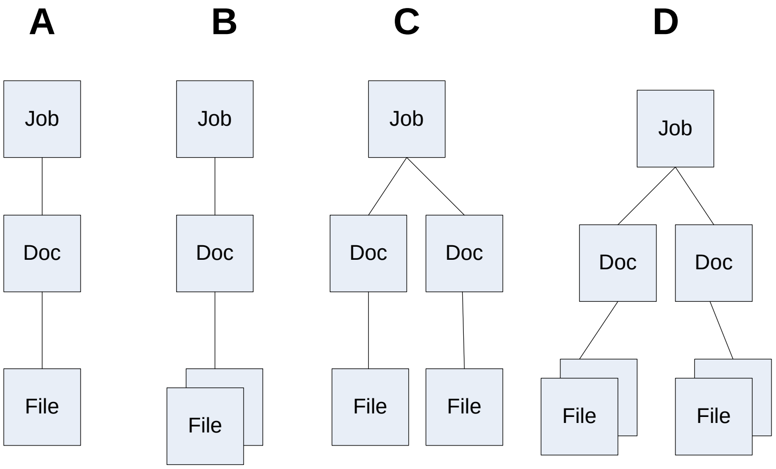
Discussions of “Walk-up Scan with Pre-Created Scan Job Template” use case –


(12b) RepostorySpecificGet
(12a) RetrieveScanDocument(User,JobID, DocID)
Platen
(8) Scan Document Data and store doc.
(5) CreateJob(JobTemplate)
Scan Client
(11a) EOJ event
Scan Service
Document Repository
(11b) GetJobElement(status)
(10) Update Job Object @ completion
(6) Scan Job is created and queued
(9) Update Doc Object with doc location(s)
ScanJob
- Ticket
- Description
- Status
(1) ListScanTemplate(Username)
(3) Scan Template
(4) Modified
(2) GetScanTemplate(TemplateName, UserName)
(7) Scan Document is processed
Scan Document
- Job Receipt
- Doc Receipt
- List of Doc
Template Service
Template Repository

* Note: When Scan Client is initialized, it discovers its target Scan and Template services by:
- WS-discovery
- Configuration of Scan Client
- LDAP
- UDDI

- The diagram intents to illustrate the operational semantics among a Scan Client, a Scan Service, and a Template Service. All services don't have to reside in the same device. The Scan Client could be local or remote to the Scan Service.
- A template is syntactically the same as a ticket. It contains originating user name, site-related policy indicating the need of security between the Scan Client and the Scan Service.
- Semantic requirements of the Scan Service by walking through this use case:
  (0) For the following steps, it is assume that the Scan Client has already discovered or configured all reachable Scan Services and Template Services and the user has already selected the target Scan Service and Template Service to use.
  (1) For the user to select a template, the Scan Client issues a ListScanTemplate (RequestingUserName) request to Template Service which returns a list of scan templates that the user is allowed to see according to the user's site policy. The user is authenticated for the right to access the list of templates a priori.
  (2) The user selects the desired template using GetScanTemplate(TemplateName, UserName). The UserName can be replaced with the most trusted name of the user's security domain.
  (3) The user obtained the default template returned by the Template Service.
  (4) The user fills out the template, and the template is modified.
  (5) The user pushes the green button: the Scan Client issues CreateScanJob (Template, UserID,). JobID is returned.
  (6) The Scan Job is created which consists of Job Ticket, description elements, and status. (The description element not only has RequestingUserName, but may also contain JobAccountingID for job charging purpose.)
  (7) Ticket is applied to create Document object. If "Must Honored" is specified in Scan Job, the Scan Service does not create the job unless it has all capabilities to process the "Must Honored" features of the job. Otherwise, the Scan Service performs its best effort to create the job and the associated documents. There should be a JobReceipt that records the actuals used to process the job. The return data of a successful CreateScanJob from the Scan service shall contain: JobID. A failure due to Must Honor conflicts returns unsupported elements or values that could not be. When the Job creates the scan document(s), the Scan Service must also process the "Must Honored" features of the document(s). Therefore, each document created must be populated with a document receipt that records the substituted value of processing elements when "Must Hornored" is false. Because the processing instructions could be different from one document to another, especially if there is a job instruction sheet between documents. This will also provide information for actual data for job accounting per document. In order to report error back to the user when "Must Hornored" is true and the Scan Service can not satisfy the request, we need a new "Job State Reason" to inform the user why the job has an error in this case. This type of error tends to occur given the fact that in XML schema, there is no way to specify intra-element dependencies, e.g. must be black-and-white and 9600-dpi; there is no way for the Scan Client to specify the constraints in the CreateJob request.
  (8) Document data is scanned and stored at scan destination by the Scan Service.
  (9) Populate document location for each document in the Document Object.

- There was a lengthy discussion on Job, document and image file. We discussed boundaries for a collection of images that comprise a document and whether we need multiple document objects. In summary we concluded that the following types of documents and files should be allowed:



A. Single Document Job: One document object that contains one single file. There is one file URL for the single document location.
B. Single Document Job: One document object that contains multiple files. Each image in the document data is stored in a separate file. There is one file URL per image file which constitutes the document.
C. Multi-Document Job: Job object contains multiple Document objects. Each document can have different set of processing parameters from another. Each file contains a specified number of images for the document. Each such set of images is stored in a separate file. The document object contains one file URL for the image file location.
D. Multi-Document Job: Job object contains multiple Document objects. Each document can have different set of processing parameters from another. Each image in the document data is stored in a separate file. The document object contains a file URL for each image file location.

Note: A Scan job has 1~n document objects. A Scan Job has a unique ID within a unique Scan Service. The documents are number 1 to N within the Job. The scan destination of a multi-document job is a folder for multi-document files. After the document file is stored, the document file URL will be updated in the document object as the document location.

Note: The job receipt provides job-wide instructions of what have honored for the job. The document receipt provides what document processing instructions were honored for the document.

(10) Update Job status at job completion.

(11) The Scan Service notifies the Scan Client the scan document data is available by one of the two ways below:

a) EOJ event is sent to the Scan Client, or
b) The Scan Client checks EOJ status by GetJobElement(status) request.

(12) The Scan Client retrieve scan document by one of the following two ways:

a) Send the Scan Service a RetrieveScanDocument(User, JobID, DocID) request, (Scan data is PULLED from the Server) or

b) Send a Repository specific GET request to retrieve the document from the Document Repository using the document file URL. (Scan Data is PUSHED by the Server)

Discussion of how the Scan Service should serve the retrieval of scan document data -

- Should we allow a user to retrieve Scan Document at document level? Or at page (image) level?
- We had a lengthy discussion of how scan image is placed in a document or a file.
- Consensus: In a scan job, a document can be defined by placing an instruction sheet in front of a group of images. So there are multiple documents, each document can be a single file or multiple single-image files. When a Scan Service assigns the filename, the name should have JobID, DocID, and PageID. When the Scan Client asks for retrieval of scan document by specifying the JobID, all document files for the job shall be returned by the Scan Service. If the JobID and DocID are specified, the Scan service shall return the specific document data. If JobID, DocID, and PageID are specified, then the specific image(page) shall be returned *only when the page(image) was stored as an individual file*. In other words, only documents or images stored as files, each could be a document consists of multiple pages or a single page of a document, can be retrieved from the Scan Service. This semantics will allow the Scan Service to use MTOM protocol to transmit each file within a job, be it a document or a page of a document, as a MIME part to the Scan client without needing to identify the page boundary when the client asks for a specific page. It is the client's responsibility to retrieve a specific page from a document file with multiple pages. The Scan Service will record all file URLs for each job as an ordered list of scan destinations which can be retrieved by the client to locate document data files of the job.

Action Items –

- Pete/Nancy to provide MFD meeting minutes
- Pete to investigate whether Xerox can sponsor a face-to-face MFD meeting in November in Rochester before the December face-to-face meeting.

Next Teleconference: Oct. 4, 2007 Thursday 3pm EDT to summarize the next steps. Plan is to get all updates for the working draft and schema/WSDL ready for the teleconference on Oct. 11th.