Meeting Minutes

PWG MFD Semantic Model Face-to-Face Meeting
July 10, 2007

On-Site Attendees:

Peter Zehler	Xerox	Peter.Zehler@xerox.com
Nancy Chen	Oki Data	nchen@okidata.com
Jerry Thrasher	Lexmark	Thrasher@lexmark.com
Dave Whitehead	Lexmark	david@lexmark.com
Lee Farrell	Canon	Lee.ferrell@cda.canon.com
Takashi Nakamura	Kyocera	Takahshi.Nakamura@ktd-kyocera.com
Koji Kubono	Kyocera	Koji.Kubono@ktd-kyocera.com
Grant Gilmour	366 Software	grant@366software.com
Chris Story	Ricoh	Chris.Story@ricoh-usa.com
Ben Kuhn	Microsoft	benkahn@microsoft.com
Mike Fenelon	Microsoft	Mike.Fenelon@microsoft.com
Erhan Soyer-Osman	Microsoft	erhanso@microsoft.com
Andrey Savov	Toshiba(TABS)	Andrey.savov@tabs.toshiba.com
Jason Wei	Toshiba	Jason.wei@tabs.toshiba.com
Rick Landau	Dell	Richard_landau@dell.com
Harry Lewis	InfoPrint Solutions	harryl@us.ibm.com
Jan Walter	Product Architect	jwalter@peerless.com
Ron Bergman	Ricoh	Ron.Bergman@ricoh-usa.com
Craig Whittle	Sharp Lab of America	cwhittle@sharplabs.com
Ole Skov	MPI Tech	osk@mpitech.com

Phone-In Participants:

Ira McDonald, Blue Roof Music/ High North Inc.
Glenn Petrie, Epson

Collaboration on MFD between Microsoft and PWG –

- PWG Chair Harry Lewis opened the session with thanking Microsoft sponsoring this meeting that has been very productive so far. PWG always strives for building a standard semantic model that suits the industry across all vendors' products. Recently PWG has started specifying a standard MFD semantic model that will be independent of all different platforms and interoperable across all vendor products. It has been recognized that Microsoft is modernizing its operating systems, sharing the common interests with PWG in advancing technology. PWG would like to have an open forum to discuss collaboration with Microsoft in technical aspects. No intellectual properties or product schedule will be asked to disclose.
- MFD Chair Peter Zehler provided a high level overview of MFD modeling that led into a discussion of collaboration with Microsoft in scan service:
    - PWG semantic model version one is basically a model from internet printing protocol. Right now PWG is working toward a MFD semantic model.

- The PWG semantic model has a container which is basically a server. The server has a "system" element that provides summarized total of services. "**S**ervices" element has print, copy, scan, emailin, emailout, faxin, faxout, transform (transform data later used in workflow). The device aspects of the server has console, cover, finisher, interpreter, marker, … and some aspect of WIMS. WIMS has the concept of managers and agents, and the view of devices not subunits so that free-standing devices can also be managed.
- The focus of today's discussion is service aspect of MFD model. Print service is done. PWG had started on defining fax service, but it has been recognized that both print and scan services have many common properties and components that can be used for many other MFD services such as fax, copy. Thus the focus now is on Scan Service. There have been a lot of discussions on how to keep the industry from diverging the semantics of scan service. Microsoft has published WSD-print and WSD-scan that have heavily reused the PWG semantic model. Because of the reuse of the common model, it was quite trivial to build gateways to UPnP, IPP, WSD, JDF-print,… ,etc. PWG would like to see the reuse of the semantics for scanning and discuss with Microsoft on collaboration of the reuse of PWG semantics around the scan service.

- Collaboration Discussions:
  - Mike Fenelon explained that Microsoft's concern is its contractual agreements with individual company's contribution to WSD-Scan. Unfortunately Microsoft can not provide WSD-scan specification as a whole for the basic starting point of PWG scan service. But Mike is free to discuss the basic concept of scanning, and contribute according to his own knowledge to move the model to the right direction. Cut and paste the content of WSD-scan specification at all level is not allowed due to many intellectual properties involved in the specification. Peter Zehler reported that Xerox has allowed him to provide PWG his inputs contributed to Microsoft WSD-scan before. After some discussions, the group's understanding was that Microsoft (Mike) is happy to contribute as a PWG member, but can not let PWG take the entire WSD-scan specification due to the contractual agreement with many companies. But there is nothing to prevent an individual company to take whatever it contributed to WSD-scan before to PWG scan service if the company chooses so. Mike recommended all PWG members continue working on MFD scan services, he is happy to contribute along with all members to keep Scan services and WSD-scan aligned.

MFD Scan Service Schema Walk-Through

- The schema file for MFD Scan Service to be discussed will be available soon at the PWG MFD web site after successfully complied by the .NET WSDL compiler. Currently it has successfully run through the Jigsaw XML complier.
- The PWG semantic model has multiple services, associated with each is a queue and a set of elements such as status information, descriptive elements (settable or read only, controlled by policy), processing elements (defaults, ready, supported, and a place holder for extension), and status elements inherited from the imaging services class. There are state messages and state reasons. State messages are localized state reasons. The only scan service specific status extensions are the scan service specific counters. Both the status requests and return of state messages are in the current locale of the server – like IPP. It is possible that there will be requests made in five different locales, each is returned with a state message that is in the same locale of the request. There is a place holder for vendor's extension for a state message. The extension of scan specific descriptive information is done the same way. Scan service has subunits which provide the view to the subunits that the services depend. Similar to print there are job level processing elements. Currently the group has agreed to the high level semantics, not lower level

yet. Each service has a queue which has a list of jobs. Each job currently has a single scan document. More elements will be added into the schema as more are defined in the Scan Service specification.

- Scan Service Operations proposed in the latest specification not only have scan service specific operations, but also operations common to all devices such as disable/enable, pause/resume. Disable/enable has no parameter and returns successful or failed status. Many parameters in operations still need to be defined. Currently in web service binding, there are two port types implemented: one handles scan service specific operations, another port type handles template management operations such as store, retrieve, list templates. There are also operations for get, set, and store templates, yet the elements of the template has not been defined. The CreateScanJob is an operation to send request to scan service, the details of which still need to be defined. The destination of a scan resides in the scan job ticket which is a copy or a reference of a scan job template. A scan job template is an unbound ticket that can be predefined and store away. When a scan job is requested, the template is copied into the scan job ticket which contains all job processing parameters, one of the elements is the scan destination. The return of a CreateScanJob operation is a list of elements that wasn't supported in a JobId. The job from CreateScanJob request has a description and a list of document processing instructions which is inherited from the parent imaging job class. Right now the imaging job class does not have scan specific extensions.

Use Cases Review

- The use cases previously discussed in weekly working group meetings were reviewed to see how data is exchanged between client and server.

  - Walk-Up Scan with Pre-created Scan Template:

    Step 1: User place document on scanner

    Step 2: Scan Document operations:

    User uses scan service UI to invoke the ListTemplate operation that allows a user to select a template from the list of stored templates, each can be identified by an Id or Name. A template has descriptive and processing information of a scan job, but does not have the metadata associated with the content of scan document. Metadata of this sort is out of the scope of scan service, but can be described in transform service of MFD.

    The scan service performs a CreateScanJob operation that allows scan service to create a scan job. The operation has a parameter which is either a copy or a reference to scan template from which the scan job ticket is created and associated with the job.

    Step 4: The scan service performs a ScanJobReponse operation that returns the scan JobId to the user (who could be a network scan user). In Print, there is AddDocument for multiple document jobs, then CloseJob operation is to signal a job is ready to be scheduled. In scan, the assumption is there is only one single document per job. Therefore there is no need to AddDocument and CloseJob. There is the use case when a user would want to scan a bunch of photos, each should be output as a single file or document in JPEG format. This case can be viewed as single job with single document object association with which multiple documents can be output. It is a hardware specific implementation issue as whether each sheet requires a push on scan button. A question was asked whether we should consider a complex use case when a user would want to be able to perform a scan by instructing certain originals be scanned from ADF, then others from the platen in some ordered sequence, and back and forth, so the scan service needs to scan and assemble the pages properly from

both inputs into a single document. It was recognized no user in the right mind can perform such complex scan. The document assembly process should be considered the function of a higher level orchestrator such as a software tool or document workflow application working from multiple jobs. The scan ticket lifecycle diagram was examined again to see whether the scan service model need to changes to accommodate these use cases. After some length of discussion it is clear that we need to allow a single job to output multiple scan documents from one document object, such as one JPEG file per scanned photo from a single ADF photo scan job. These individual document files can then be stored in an internal or external repository. Since there is no way the scan service can mandate, at the job level, what document metadata (data about the content of document data) should be associated with each document that is stored in a document repository, metadata is out of scope of scan service model. Clearly these stored image files can then be OCRed and index information be added for further use by a higher level vendor's application, but this is out side the scope of scan service. Clearly we need to show multiple cardinality association of scan documents with a single document object. It is also recognized that the document object model in the pre-standard job ticket API already allows multiple document data references. All documents in one job have the same job processing parameters. Different job processing instructions will require a separate job. Nancy Chen will take the action to add the cardinality association (0..*) from the document object to multiple document data in the scan ticket lifecycle diagram. It seems that how each image data file is output by the scan service will need to be modeled within the scan service itself.

Question was asked whether the CreateScanJob operation should be synchronous which means the operation does not complete until data is available to the user. None responded. Therefore the ScanJobReponse implies that the job is available for scheduling. It is possible a job can indicate JobHoldUntilTime so that multiple jobs are created and scheduled but on hold until the user walks up to scanner to release them.

Step 5 Scan Job is scheduled right at the implication of ScanJobResponse.

Step 6 The completion status of a scan job is determined by polling the state of the scan job. The completion status is reached when the last output of the scan job is made. There is a scan service counter which keeps how many images have been scanned.

Step 7 The alternative to polling the completion state of a scan job is by event driven model – all events will be reported in the model, it's vendor-implementation dependent for mapping to a specific eventing technology.

Question asked from Toshiba Attendee Andrey Savov: When a user would like to scan 2000 pages within one job, how can scan service allow user to continuously feed the papers into ADF or the platen? One proposal is to model after IPP that has AddDocument and CloseJob. Instead of CreateScanJob, use ScanJobResponse and AddPhysical to allow an open-ended job to add more originals to a scan job. Andrey Savov will provide detailed description of the use case for the group to consider.

- Create Scan Job Template Use Case:
  - Case 1 User Create a template through a scan client -

Step 1 User runs a template manager from a scan client.

Step 2 The template manager at scan client sends a GetScannerElements request which obtains the job description elements and the defaults of the supported job and document processing elements from the scan service, much like obtaining the DPA initial value job in print.

Step 3 The scan client composes a template.

Step 4 the scan client sends ValidateTemplate request to the scan service for checking the validity of the values in the composed template.

Step 5 PutTemplate operation stores the template in scanner or a remote template repository. The template has a URL which can be used as a reference to the template in subsequent operations. The templates can be retrieved from its repository which could be the scanner or a remote repository. For simplicity the ListTemplate request could be sent to the scan service which discovers and lists all templates stored (locally or remotely).

- Case 2 User Create a template directly with the Scan Service –

This case is same as Case 1, except that the scan client is collocated with the scan service.

Step 1 User walks up to the scan service UI.

Step 2 The scan service UI displays the supported job and document processing elements, and user selects his desired values for the elements.

Step 3 The scan service UI automatically validate user's input values.

Step 4 A template is created by the scan service using the validated inputs.

Step 5 The scan service "PutTemplate" operation stores template with its URL. Through scan service UI the user can list the stored templates and select the desired template for a scan job.

- Walk-up Scan and Send Document to Storage Use Case

Step 1 User puts document on scanner.

Step 2 User gets a list of templates from scan service.

Step 3 User selects template.

Step 4 User optionally modifies the copy of template (not the template stored in repository).

Step 5 User pushes green button. The template is copied into job ticket. A scan job is created, and the scan ticket is bound to the job. AddDocument and CloseDocument operations (for multiple document job) are no longer needed. At this point, job is implicitly scheduled.

Step 6 Document data is scanned and associated with the document object.

Step 7 Document data is stored.

Step 8 End of Job event is sent to the user.

Step 9 As an alternative to Step 8, user can query the status of scan job via GetSanJobElements request.

In the document object there is a counter for number of images, and in the job object has a counter for the number of document data. These counters are indicators of the processing status of the job.

If there is an error, a system event should be sent to user. For an error in a subunit for example, the scan service will be put in a "pause" state for error recovery, e.g. a jam recovery, which is an administrative action. At protocol level, no back-up protocol or "resume" action is required.

- Scan from PC and Store Document in Repository

- This is the same as the last use case, except the scan is done from a PC.

Step 1 User puts document on scanner.
Step 2 User runs scan application from a PC.
Step 3 The Scan application sends GetScannerElements request to obtain the job description elements and the defaults of the supported job and document processing elements from the scan service.
Step 4 User selects scan parameters. This could be done by PC which displays the user's template and let the user set the parameters by value or the user could use a template manager to select a template and modifies some parameters of the copy of the template, or simply by selecting a template by referencing the template (by URL).
Step 5 Optionally ValidateTemplate request is sent from PC to the scan service. If this is not implemented, when a conflict occurred, it's vendor's decision to resolve the conflict. In print, there is a "fidelity" attribute – 'true" means don't proceed if there is a conflict, 'false' means proceed with best effort. The question is how to return the ScanJobResults to the client? Should we tell users what were used in the job ticket? Therefore ScanJobResults should return a JobReceipt that tells the user whether the job is accepted or rejected and contains actual job ticket to be used for job processing.
Step 6 The PC sends ScanJob request to the scan service with template by reference or by value.
Step 7 Job and document is created with a copy of template in scan service.
Step 8 Scan job is available for scheduling.
Step 9 ScanJobResults is sent back to PC. One could put the job on hold as soon as job is available for scheduling. Then the job won't be scheduled.
Step 10 Job is scheduled and document is scanned by scan service.
Step 11 Scan service stores document data.
Step 12 Job is marked complete by scan service.
Step 13 End-of-Job event is sent by scan service to PC.
Step 14 As an alternative to Step 13, the PC performs status poll of the scan job.

Proposal for Some Document Scanning Terminologies –

- The scan area has X and Y orientation. Y is the height and is in the slow scan direction. X is the width and is in the fast scan direction.
- This purpose is to allow user to set the offset relative to the origin and the size of the scan area.
- We do not want to define scan area in relative to the content of the document being scanned.
- We want to be able to describe the area to be scanned in relative to the scan mechanism.
- The origin is where the scan starts, or where the scan mechanism first meets the media.

Scan Ticket Concept Discussion -

- There seems a preference to move back to the InitialValueJob concept used in DPA. The job ticket or template not only includes job processing instructions, but also descriptive elements that allow default or initial values of descriptive elements and processing elements.
- Scan Job Template has the following elements:
  - Job Description Elements

    This is the metadata of a scan job, not the metadata of the content of scan document. These are elements settable by administrator, user, or operators.
    - JobName

- JobOriginatingUser (job owner)
- JobInformation
- JobPassword
  - This is to enable a job to be held until the user walks up to scanner, puts in secret password to allow job to continue.
- JobPasswordEncryption
  - This allows user to query what kind of encryption is supported for job password encryption.
- JobAccountingID
  - This is the scan service charge account number.

The locale of description elements is supported in the current locale of the server which is the locale of a request operation. No translation from one locale to another is defined.

- Job Processing Elements
  - JobHoldUntilTime
    - Absolute time or duration (need to be qualified in semantics)
    - There is also a ReleaseJob operation to release a job on hold
  - JobMessageToOperator
    - Display a message to operator when a job is processed.
  - Note the following elements should be removed from the ImagingJobProcessing super class -
    - JobPriority is the priority for a scan job to be scheduled, i.e. the scheduling priority among scan jobs. This element should be removed, because there is a need to match the actual scan media to a scan job. User needs to make sure the right media is placed for the scan job. There is no way for scan service to automate this matching in order to support priority.
    - JobSaveDisposition (print-specific, save the job for later processing such as a demand re-print)
    - JobRecipientName (to denote who is the recipient for output in print. In scan we have data store destination.)
- Document Processing Elements (elements control the actual scanning)
  - AutoExposure
  - Color parameters
    - Encoding
      - AutoDetect (binary, grey, color)
        - Based on the value of encoding, Color space, samplesPerPixel, BitDepth will be affected and honored if specified.
    - Color Space
    - SamplesPerPixel (implied by color space)
    - Bit Depth
  - Contrast – need description
  - Brightness -
  - Magnification – expressed in X (height), Y (width)
  - Gamma – a measurement of contrast and brightness
  - Saturation
  - Resolution (X, Y, pixel per centimeter, prefer to see an absolute unit)
  - Rotation (enumeration 0, 90, 180, 270)

        - Sharpness
        - Filters – need names with semantics, common filters, some applies to transform service.
    - Scan Job Status Elements
        - JobId (An unique integer that cannot be reused for a reasonable period of time)
        - JobState (pending, processing, processing stopped, pending held, completed)
        - JobStateReasons (0..N, e.g., completed with success/failure)
        - CreationDateTime (can goes down to seconds)
        - CompletionDateTime
        - NumberOfDocuments
        - ScanCounters
            - Images – an electronic representation of a media sheet side
            - Koctects – the total size of scanned image data

Related Notes from Closing Planary:

- The weekly MFD Model teleconference time has been changed to Thursday 12pm Pacific time (3pm EDT) to accommodate some west coast members' conflict in schedule.

Action Items:

- Nancy Chen and Peter Zehler will update the scan service specification and XML schema according to the results of today's discussion and post them to the group as soon as possible.
- Andrey Savov of Toshiba will provide the detailed descriptions of their open-ended scan job use case for the group to consider.