Image Validation in End-to-End Workflows

Peter Krogh; Senior Project Manager, dpBestflow Project, American Society of Media Photographers, Philadelphia, PA, USA

Abstract

One of the most vexing issues in digital imaging is the challenge of image validation. Images may be corrupted at any point in the handling chain, from capture through transfer, image editing, storage and migration. Our research into image validation workflow has led to a lifecycle-based set of recommendations, used in conjunction with the DNG file format. These techniques can create an end-to-end workflow that is validated at each step, once an initial visual verification has been performed. This leads to both an increase in security, as well as a reduction in the resources needed to maintain the integrity of image files.

The traditional approach to data validation is to make a database of checksums of stored files, and to run a periodic validation sampling. While this approach does provide some good protection for static archived files, it is only appropriate for files that are completely static - any alteration to the file, such as image adjustment or the addition of embedded metadata produces a mismatch. Moreover, a traditional checksum approach typically relies on an external database of checksums, which creates a difficult workflow as files are transferred between different computers.

This paper provides an alternate methodology for a fully validated image file workflow, from initial image creation through to archive. It makes use of two tools to accomplish this: the Adobe DNG file format, as well as Parametric Image Editing (PIE) software. In a Parametric Image Editing environment, source image data is never modified, but instead is reinterpreted. This allows preservation of the original image, even as the image may be re-rendered according to different parameters. The Adobe DNG file format includes an area to store the source image data, rendering settings, metadata, as well as one or more fixed renderings of the image. One of the metadata fields that is part of the specification is an open source MD5 Checksum that refers only to the unchanging source image data.

Effective use of the DNG file, therefore, creates a portable validation key that can be assigned very early in the lifecycle, and travel with an individual file. The checksum remains viable even as a file is changed, or as the image is readjusted. Adobe has also released several free software tools that can check on the integrity of large collections of image file data automatically and reliably. These tools can, for instance, reliably identify a single changed bit in a file inside a multi-terabyte archive.

While the DNG can provide the majority of data validation needs for a digital image library, other validation tools are needed to fill in the gaps. Visual validation is still required at the start of workflow, and transfer validation should also be used regularly. When images must be converted to a rendered filetype, it becomes necessary to rely on the more traditional data validation tools.

Data Volatility and Validation

Perhaps the most difficult task in data validation is the verification of volatile data. It can be relatively straightforward to use a checksum to verify that static data remains unchanged. Any data that is subject to intentional user change presents a far more difficult challenge. Often times, it's simply impossible to tell the difference between a change that is desired and one that is undesired.

While differential or incremental backup schemes may help protect volatile data, they generally don't offer a simple verification path. And since incremental backups generally generate a data set many times larger than the original data, they often become unworkable for image collections.

This paper offers a method for data validation that splits volatile from non-volatile data on a sub-file level, so that it is possible to verify the integrity of the non-volatile component of image data that may still be in a state of flux, with regard to user interpretation of the image, or any descriptive or organizational metadata.

Data Lifecycle

The foundation of workflow construction is an understanding of data lifecycle. We use these concepts to help illuminate data handling throughout workflow, and that includes validation practices. We divide image workflow into four sections - *Capture*, *Ingestion*, *Working* and *Archive*, as shown in Figure 1.

- *Capture* includes the process of creating or scanning images. These may be scanned TIFFs, digital camera JPEGs or proprietary raw, or in-camera DNG files.
- *Ingestion* encompasses the automated set of steps that are performed to a file after capture. For camera captures, these are the download/naming/tagging/backup processes that are done when the image is brought into a computer. For maximum data verification, we suggest conversion to DNG during *Ingestion*.
- The *Working* phase is the volatile state between *Ingestion* and *Archive*. This phase may include additional processes that are desirable prior to image file archiving, or it may simply refer to the handling requirements of field-to-studio transfer. In a fully Parametric Image Editing environment, it may be possible to bypass this phase entirely, and ingest images straight into the archive.
- The *Archive* refers to a non-volatile storage status for image files. It presents, by far, the most straightforward verification challenge of any workflow phase.

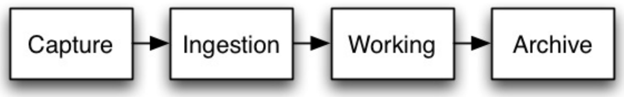


***Figure 1** diagrams the 4 lifecycle phases we use in describing workflow. Note that the ingestion phase is the optimal phase for conversion to DNG, since it attaches the checksums as early as possible.*

In the second half of this paper, we'll look more closely at the tasks associated with each of these lifecycle phases, and how validation comes into play.

Data Validation Tools

In order to create a validated end-to-end workflow, we'll need to make use of several different validation tools. Once we define these, we can see how they can be combined to maximize verification and minimize overhead.

File Structure Validation

One of the easiest types of data corruption to detect is damage to file structure that prevents the parsing of the image, since that often triggers a warning message from the software alerting the user to an error.

Visual Validation

We use this term to refer to a process of visual inspection of an image file when it is decoded and displayed by a software package. Visual validation is not a way to determine that all data in a file is pristine. Instead, this technique is used only to determine that certain gross errors have not occurred, as shown in Figure 2.

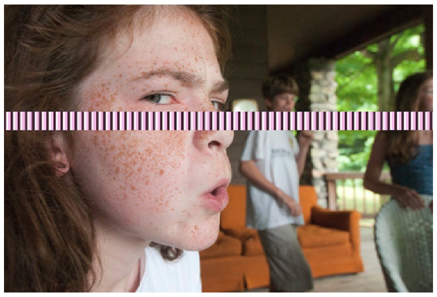


***Figure 2** Here's an image that was corrupted in download, due to a faulty hard drive bridge board. It's an example of corruption that shows up in a visual validation, but not in a check of the file structure, since the data in the image block was scrambled but did not change in length. While it's easy to catch with human inspection, there's no automated way to find this type of corruption at the present time, if it is present in the original file.*

Image File Integrity Validation

We use this term to mean that the image data has not been corrupted in any way. Generally this means that the file structure is intact, and that the image data has not been altered in an unwanted way by even one bit.

Validated transfer

Computers generally copy images files from one medium to another without error, but the safeguards built into the operating system are not foolproof. We have documented evidence of both omissions in file copying, as well as errors in image file integrity in the new files. Validated transfer can be achieved with software that has bit-for-bit verification included. It's also possible to use command-line tools to ensure that all files and all bits have been copied properly.

Completeness Validation

The final type of image verification is confirmation that the image collection contains all the image files that are expected. In order to check for completeness, it's essential to have a comprehensive catalog of what's supposed to be in the archive. A periodic check for missing items can confirm that no files have gone missing due to machine or human error.

Parametric Image Editing

Currently, the most far-reaching development impacting a validated end-to-end workflow is the advancement in Parametric Image Editing (PIE) software such as Adobe Lightroom, Apple Aperture, Capture 1, Bibble, Nikon Capture NX and so on. In this class of software, image adjustments are created by rendering engines according to the user's instructions. Adjustments are saved by preserving the instructions, rather than altering the pixel's color and tonal information. This contrasts to the function we generally find in a raster image editor like Photoshop, where the original pixel values are often modified by the software.

Parametric Image Editors have largely grown due to the popularity of raw file photography, where the capture format is proprietary and original source image generally cannot be altered in any useful way. This has profound implications with respect to backup and validation, particularly from a workflow and lifecycle perspective. Parametric Image Editing allows a workflow construct where image data can be archived immediately or almost immediately, since the source image data is inherently non-volatile.

DNG File Format

The DNG file format was built from the ground up to be used in a Parametric Image Editing workflow. It is an openly documented, openly and freely licensed format. DNG files can be created natively by some cameras, such as the Leica M8 or Pentax K10D. Other proprietary raw file image types can be converted to DNG in post-processing, while retaining all the original raw image information. (A small number of cameras, such as the Sigma SD14 that uses the Foveon Chip must linearize raw data in order to store as DNG, and therefore no longer retain all raw image data). Additionally, it's possible to convert certain rendered filetypes, such as JPEG and flattened TIFF files to DNG for use in a parametric editing environment.

While the original goal for DNG was more oriented toward providing a standardized raw file capture format, over the last few years, many enhancements have been added that are related to post-processing and computer workflow needs. Among these, as of the DNG specification 1.2, was the addition of two internal checksums that refer to non-volatile image data. Tags 50972 and 50973 are MD5 checksums that refer to the source image data and the source image file, respectively.[1] Figure 3 diagrams the relationship of the checksums to the non-volatile image data, and shows that volatile data is stored outside the checksummed data blocks.

The utility of this arrangement – splitting volatile from non-volatile data – is further enhanced by the other capabilities that are provided by the DNG format. The format provides a structure for a vast amount of descriptive, organizational, usage and ownership metadata. The specification also outlines how to embed user instructions for image enhancement and color profiles that can be used to transform source image data into a rendered output form. And, finally, the specification provides for the embedding of fixed renderings of adjusted image files, so that the finished images are portable, even to applications that can't properly decode the source image data.

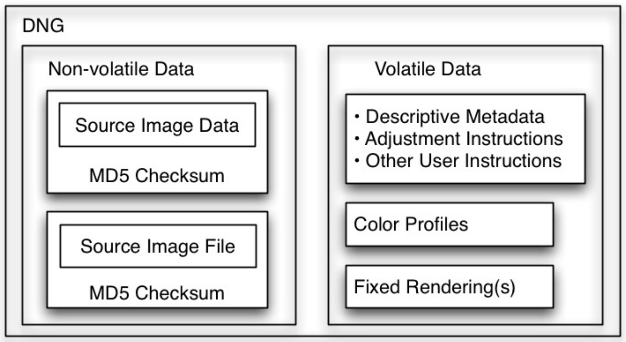


Figure 3 *shows a simplified diagram of the contents of a DNG file, according to the 1.3.0 Specification.[2] The source image data and the source image file (if present) are assumed to be non-volatile data, and an MD5 checksum is computed for each. Other data, such as the descriptive metadata, profiles, and even fixed renderings are stored in the file outside of the checksummed blocks of the file.*

Note that the DNG specification 1.2 in particular has opened up the format specifically for non-Adobe software to take advantage of these capabilities. This includes additions of color profiles that are explicitly different from the ones used by Adobe [3], as well as a way to note which software may have been used to previously render the image [4].

The above capabilities make the DNG ideally suited to serve as an image file format that provides unique utility throughout the entire image lifecycle, including a file being passed among multiple computers and applications.

Validation Behavior in DNG handling

This section will outline the current behavior of DNG handling tools, with an emphasis on the behavior of Adobe software. [5]

DNG Checksum Creation by Cameras

At the moment, no manufacturer writes the checksum into DNG files created in-camera. We would like to see this changed in the future.

DNG Checksum Created by Adobe Software

As of the DNG specification 1.2, released in spring 2008, all current Adobe software writes the Source Image and Source File checksums into any DNG files that are created. These checksums are only recomputed if the user chooses to linearize the DNG for backwards compatibility. Otherwise, they are passed along just as any other file metadata, and can be used to verify that the non-volatile image data remains unchanged.

Image Data Integrity Verification by Adobe Software

Adobe software that is DNG version 1.2 or later compliant will use the checksum to verify that image data remains uncorrupted whenever it parses the source image data.

- When a DNG file is opened into Photoshop through Adobe Camera Raw, or into Adobe Lightroom's Develop Module, the checksum is verified, and the user is presented with a warning in the event of a mismatch, as shown in Figure 4.
- Whenever a DNG file's embedded preview is updated by Adobe Lightroom or Camera Raw, a verification check is performed.
- Additionally, all DNG files sent through Adobe's free DNG Converter software will be verified. We'll see more of this software in the next section.

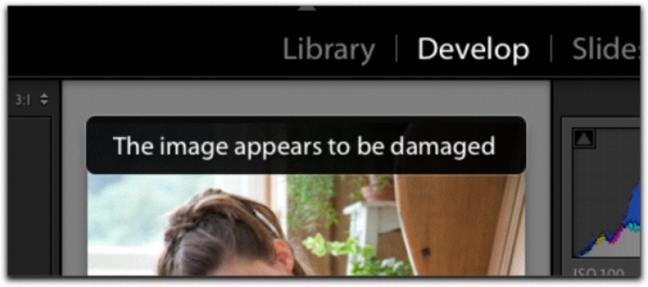
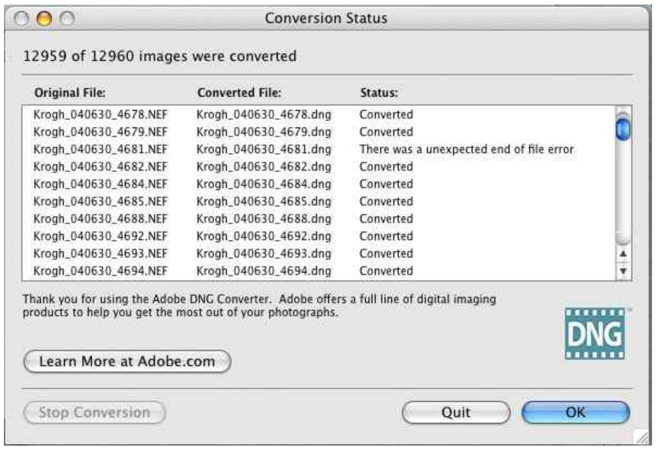


Figure 4 *shows the error message provided by Adobe Lightroom when it encounters a mismatch in the embedded and the computed checksum.*

File Structure Verification upon DNG Creation

Whenever DNG files are created, the image file must be parsed, and the user is presented with an error message if one or more files can't be opened. This makes the free Adobe DNG Converter one of the easiest and most useful tools for verifying the integrity of raw file structure for a large number of images. Figure 5 shows an error message generated by a single corrupted raw file in an archive of more than 12,000 images.



Conversion Status

12959 of 12960 images were converted

Original File:	Converted File:	Status:
Krogh_040630_4678.NEF	Krogh_040630_4678.dng	Converted
Krogh_040630_4679.NEF	Krogh_040630_4679.dng	Converted
Krogh_040630_4681.NEF	Krogh_040630_4681.dng	There was a unexpected end of file error
Krogh_040630_4682.NEF	Krogh_040630_4682.dng	Converted
Krogh_040630_4684.NEF	Krogh_040630_4684.dng	Converted
Krogh_040630_4685.NEF	Krogh_040630_4685.dng	Converted
Krogh_040630_4688.NEF	Krogh_040630_4688.dng	Converted
Krogh_040630_4692.NEF	Krogh_040630_4692.dng	Converted
Krogh_040630_4693.NEF	Krogh_040630_4693.dng	Converted
Krogh_040630_4694.NEF	Krogh_040630_4694.dng	Converted

Thank you for using the Adobe DNG Converter. Adobe offers a full line of digital imaging products to help you get the most out of your photographs.

Figure 5*. The DNG converter can confirm file structure integrity for thousands of raw files with a single command. In this case, it found the one bad image in a group of 12,960.*

Catalog-based Workflow

In addition to the volatile/non-volatile split that DNG enables on a sub-file level, it's easily possible to split these classes of data on a file level when working with a robust catalog application. The *Archive* storage can safely preserve the image data. The catalog document that lives in the *Working* storage can be the main repository for the ever-changing metadata associated with files that have already been archived. This allows the collection manager to apply the preservation and validation tools most effectively and economically to archived images. The vast majority of the data can live in a static environment, which offers better economy and security. And the much smaller set of volatile data - adjustment settings and descriptive metadata - lives in a more expensive environment that offers robust rolling backup and disaster recovery protection, as shown in Figure 6.

The ratio of metadata size to source image data size is often in the neighborhood of 500:1.

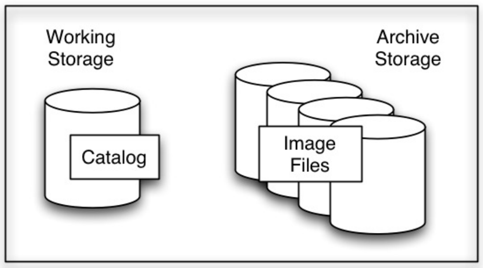


Figure 6 *An image catalog can live in the more expensive Working file storage, while unchanging source image data can live on less expensive (per TB) and more secure Archive storage.*

Data Validation Workflow

The second half of this paper presents methods for constructing a safe and efficient workflow using the tools outlined in the first half of this paper. The tasks will be divided into the lifecycle phases outlined at the start of the paper.

Fully Parametric Workflow

If all work to image files can be done by a Parametric Image Editor that supports DNG 1.2 checksumming, it's possible to create workflow that is validated at each step with a high degree of confidence, and a low administrative overhead.

Capture

As stated earlier, there is currently no camera that writes a DNG with the checksums embedded. Therefore, the only validation possible is a visual one that confirms that the image contains the subject matter and quality desired.

Ingestion

Figure 7 outlines an ideal ingestion workflow.

- Images are downloaded from the camera.
- Images are converted to DNG at the first possible opportunity and tagged with the Checksum. The original source file can be embedded if it's desirable to retain it.
- The converted DNG is backed up twice. If the ingestion is happening in the field, these can be dedicated ingestion backup drives. If the images are being downloaded in the studio and the long-term backup is accessible, then they can be backed up right away onto that media.
- The images should be cataloged for future verification of completeness.
- And finally, the images should be visually validated to determine that no gross error has occurred. If this inspection occurs after DNG conversion, it's the only time a visual validation will ever be required, except in the event of a checksum mismatch warning.
- Once images have been visually inspected, they may be erased from their *Capture* medium.
- All transfers of files should be validated transfers, if possible.
- In addition to the steps shown on the chart, it's frequently desirable to rename files, add bulk metadata and parametric default settings.

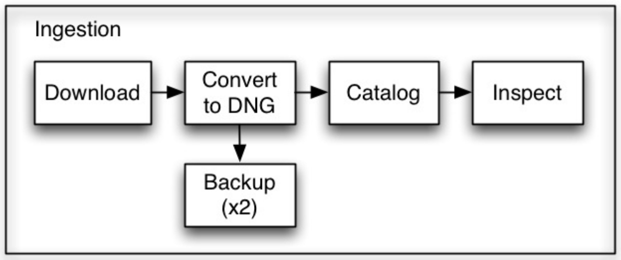


Figure 7 *shows the elements of the Ingestion process.*

Working

The *Working* phase is not always necessary in a fully parametric workflow, particularly if the files can be ingested directly into the *Archive*. Oftentimes, however, it's desirable to do some work to the files before archiving them. Figure 8 shows them in descending order of importance. The first two steps are more important than the second two, in terms of *Working* file preparation.

- It's most important that images get a permanent name attached prior to archiving. If it's necessary to rename, this should be done at this point.
- If you are going to cull images prior to archiving, this is the time to do it to prevent orphaned files in the archive and archive backups. (Note that some people like to cull before renaming, if they wish to have a name sequence with no gaps.)
- Attaching a good proofing rendering to the archived images is a useful task, but not necessary in a full parametric DNG workflow, since rendering can be updated without invalidating checksums.
- Likewise, attaching good basic metadata can be useful, but is not really necessary at this point.
- During the entire *Working* phase of life, images should be well backed up, since they are in a state of flux, and therefore innately more susceptible to corruption. We suggest both a rolling backup and a disaster-recover backup be created.

- Once images have been backed up in their *Working* form, they may be erased from their *Ingestion* medium, if it's different from the *Working* medium.
- If DNG 1.2 or later Adobe software is being used, then images will be validated in the background nearly anytime a change is being made to the image data, and whenever a DNG's preview is updated. If there is any suspicion of malfunction in the computer system, it's simple to validate by sending all working files through the DNG converted and checking the log for errors, as shown above in Figure 5. Likewise, if there is any suspicion that the collection is incomplete, the catalog of files that was made in the *Ingestion* phase can confirm or refute the loss of files.
- Anytime a file is transferred for one medium to another, it should be done by means of a validated transfer.

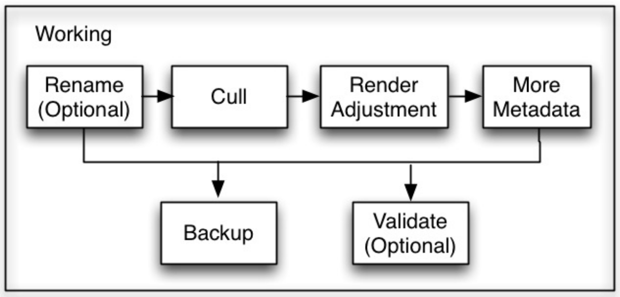


Figure 8 *shows common tasks for Working files, prior to archiving. They are in priority order from left to right.*

Archive

In a fully parametric workflow, images may be archived long before they reach their final form in terms of rendering adjustments or descriptive metadata. So it's important not to think of the *Archive* as an end-state for the data, but rather a long-term steady-state for the non-volatile data. The following list describes the processes for archiving images, as shown in Figure 9.

- The transfer to the *Archive* should always be performed as a validated transfer.
- It's wise to confirm that all expected files have arrived on the archive media before deleting them from working storage. This is generally a simple function performed by the catalog software – re-point the catalog and search for missing files.
- *Archive* images get a more permanent backup, outlined in the next section.
- The image archive should get periodic validation on several levels, outlined below.

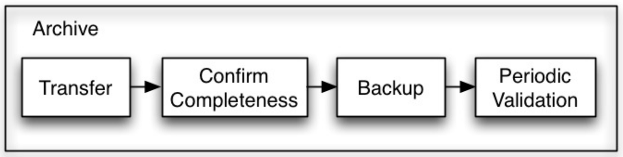


Figure 9 *The process of archiving files should follow this work order*

Archive Backup Subsystem

The *Archive* backup, unlike ones that are made in the *Ingestion* or *Working* phases can offer additional layers of protection. It's possible to add both off-site and write-once protection to the system. (Our rule of thumb is 3-2-1: 3 Copies, 2 Media, 1 off-site copy). Unlike *Working* files, which must be protected with both rolling backups and disaster recovery copies, the *Archive* backup system can be maximized primarily for disaster-recovery protection. Figure 10 shows the setup for this kind of system.

- A designated Primary copy can remain online and accessible.
- An Off-site backup can be made and either written to write-once media such as optical disk, or written to rewritable media like Hard Disk or Tape and treated as a write-once copy.
- The write-once copy can include folder-based checksums that provide validation for all data in the entire set of files. This provides clear early warning of media failure in the disaster-recovery copy.
- The Near-line backup can be write-once or rewritable, depending on whether the collection manager wishes to emphasize protection of embedded volatile data, or non-volatile data. (This decision will often be dependent on the confidence of the accuracy and integrity of the catalog document.)

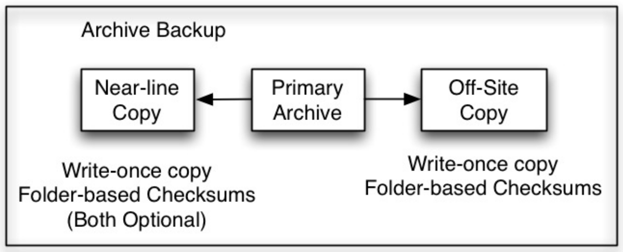


Figure 10 *shows the relationship of the primary copy of the archive with the backup copies.*

Periodic Validation

In order to preserve the integrity of the image archive, it's important to periodically verify the media and the files. The objective is to get early warning to signs of failure before any kind of data loss, and, more importantly, before there is any coincidental loss of integrity in the backup files.

The schedule for validation is determined by resources and tolerance for risk. The longer one goes between complete validation, the more risk is incurred. Of course, verifying a large archive can take time and attention, and is not cost-free.

At minimum, we suggest that both the primary copy of an archive and some portion of the backup files should be verified twice a year. Validation processes are significantly easier on hard drive-based archives, since it's generally possible to submit large groups of images to a verification queue in a single command.

Running a verification of the DNG checksum is an excellent way to do a media integrity confirmation, since it reads all the bits on a drive (if that drive is used exclusively to store DNG files).

This helps disk utilities and SMART reporting to find problem areas on a drive.

- DNG file verification can be accomplished by sending a group of images – even several hundred thousand at once – to the Adobe DNG Converter for re-conversion. The Adobe DNG Converter will log all errors, and images that have thrown an error can be inspected to see if the image data is visibly corrupted.
- The free DNG SDK available from Adobe can be integrated through command-line interface into other programs to help automate the process.
- Catalog software should be used to confirm that the image archive is complete, and contains all expected files.
- It can also be useful to run disk utilities to confirm that the volume structure and media have full integrity. These are often more informative after a DNG verification, since many of the data blocks will be read in the DNG verification process.
- Write-once media that has whole-file checksums can be verified with an even higher degree of certainty, since the process confirms both the source image data, as well as all other bits in the file.
- In many cases it is impractical to fully validate backup data on optical disc or tape. Whole-file checksums can be sampled for any data decay, and provide early warning of any errors in the storage.

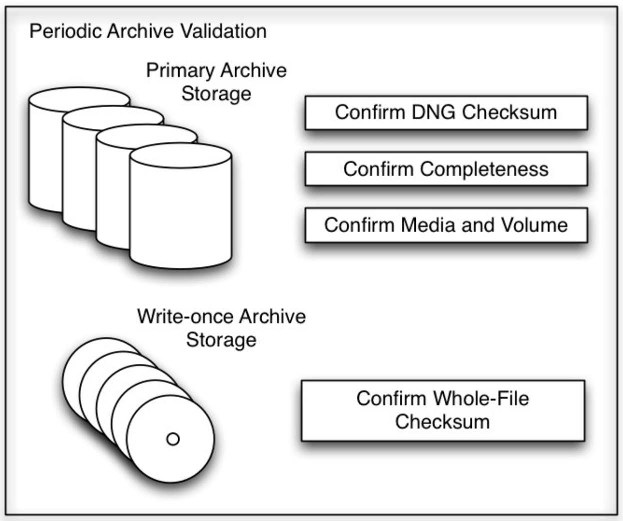


***Figure 11** The primary storage validation tasks are more rigorous than those that are needed for write-once media.*

Validation Workflow for Archive Backup Update

Updating *Archive* backups to match the primary copy of the data should be done with an understanding of the risk involved. Any system that updates a backup to match the current primary copy is inherently removing a significant amount of the protection afforded by that backup. Any silent corruption, accidental deletion, virus or media errors can be propagated from the primary, overwriting a valid copy of the backup with corrupted data.

If backups are going to be updated, it's best to verify the primary copy of the data immediately before running the update, in order to minimize the risk of replacing good data with bad.

Adding Rendered Filetypes to the Workflow

While parametric image editing tools are improving constantly, it's not always possible to use them for all necessary image editing. Sometimes a master rendered file must be created with the tools in Photoshop or other image editor that does not support a DNG workflow. Adding rendered filetypes, such as layered TIFF and PSD files to the workflow needn't create too much difficulty however. Here is some guidance on how to do this.

1. An image workflow that branches into rendered filetypes does not have to alter the parametric workflow that has been established for DNG files. The DNG files may still run through the workflow in the same way as they do in a fully parametric workflow.
2. When a master rendered file is necessary, that image file should be opened as a *Working* file from the DNG (whether the DNG has been sent to the *Archive*, or still remains in the *Working* storage).
3. The Master file should remain as a *Working* file until it is ready for steady-state archiving.
4. Rendered files may be archived similarly to the method described for DNG, but the only version of the file that can be validated with certainty are ones that are stored on write-once media.

Watch these processes in action

The procedures that are outlined in this paper are also outlined on the dpBestflow.org website. On that site, which is free and open to the public and Creative Commons licensed, you can see movies of many of these processes in action. You'll find them under at the following address:

http://www.dpbestflow.org

References

[1] The Adobe Digital Negative (DNG) Specification 1.2.0.0
[2] The Adobe Digital Negative (DNG) Specification 1.3.0.0
[3] The Adobe Digital Negative (DNG) Specification 1.2.0.0, Pages 52-53
[4] The Adobe Digital Negative (DNG) Specification 1.2.0.0 , Pages 54-55
[5] Thomas Knoll, engineer, Adobe Systems Inc.

Author Biography

Peter Krogh is a professional photographer, author, lecturer, consultant, filmmaker, information architecture theorist, and father of two. His book, The DAM Book, Digital Asset Management for Photographers (O'Reilly 2005 and 2009) has helped photographers and collection managers worldwide get control over their image collections.

The work in this paper, in large part, reflects the work he has done with Richard Anderson as part of the NDIIPP-funded dpBestflow.org project, created under the auspices of the American Society of Media Photographers, ASMP. The author wishes to thank Richard Anderson, ASMP, the Library of Congress and the entire dpBestflow team for assisting in the development and publication of this material.