



Application Note

Implementing Proofing, SoftProofing and Proof Approval in JDF 1.2

Abstract

This application note describes the rationale behind the decision of deprecating the *Proofing* and *SoftProofing* processes in JDF 1.2. It also describes how proof printing, soft proofing and approval can be described in JDF 1.2 using combined processes and how to map the deprecated proofing resources to JDF 1.2 resources.

This document focuses on the definition of proofing and soft proofing at the process level. It does not address the definition of user proofing requirements at the intent level.

Albert Such
Hewlett-Packard
alberto.such@hp.com
edition 1.02



Table of Contents

1. A BRIEF INTRODUCTION TO PROOFING AND SOFT PROOFING	3
2. PROOFING IN JDF 1.2	3
3. COMBINED PROCESSES FOR PROOFING.....	4
4. COMBINED PROCESSES FOR SOFTPROOFING	6
5. CONSIDERATIONS ON JDF ATOMIC PROCESSES	7
5.1. COLORSPACECONVERSION	7
5.2. INTERPRETING AND RENDERING	7
5.3. SCREENING.....	7
5.4. IMAGESETTING/DIGITAL PRINTING.....	7
5.5. APPROVAL	8
6. MAPPING OF DEPRECATED <i>PROOFING</i> AND <i>SOFTPROOFING</i> RESOURCES.....	8

1. A Brief Introduction to Proofing and Soft Proofing

Before we start describing the implementation of proofing (paper and soft) in JDF 1.2, first we have to have a common understanding on what proofing means and introduce some terminology that will be used in the next sections:

- **proofing** is the process of producing a printed output on a device (**proofer**) that emulates, as much as possible, the printed output that would be produced on another device (**press**) where the final production of the printed material should be performed.
- **soft proofing** is the process of displaying on a **screen** an image that emulates, as much as possible, the printed output that would be produced on another device (**press**) where the final production of the printed material should be performed.
- **approval** is the process of approving or rejecting the printed proof (proofing) or the image displayed on the screen (soft proofing). In the approval process, comments and annotations may be added to describe the reasons of the approval/rejection and to give instructions on what changes should be done to improve the result.

Note that in these definitions, I use the term press in a generic way to refer to the final production device. In this context, a press may be a conventional press, a digital press or any other device that is going to produce the final product.

An important consideration is that the original input file(s) has(have) to be processed to be printed on the final press (interpreting, rendering, screening, color management....) and also has to be processed to be printed on the proofer, that has different characteristics. The decision on which of the processing steps will be common both for printing on the proofer and on the press and, consequently, can be executed only once; and which processes are different for both devices and must be executed twice will depend on the characteristics of the proofer device, the user and workflow requirements and the type of proofing required (grayscale/color, contone/halftone, page/imposition, soft/printed...).

Thus, a proofing solution must be designed taking consideration of the proofer characteristics; the goals in terms of consistency between the press and the proofer and the different workflow configurations that it is going to support.

2. Proofing in JDF 1.2

In JDF 1.1, proofing and soft proofing were defined as an atomic process that had as input all the parameters supposedly required to perform the proofing processing and printing in the proofer device. This approach had some disadvantages:

- **Lack of flexibility:** as explained in the previous section, it is important that a proofing solution can be integrated in different workflows; but by defining **Proofing** with an atomic process, the semantics is limited to the definition of the processes and the parameters (resources) that it can take as input. Of course, the semantic could be extended by adding new resources or parameters, but this approach would complicate even more the definition of the *Proofing* and *SoftProofing* processes.
- **Lack of control:** because the *Proofing* and *SoftProofing* processes in JDF 1.1 actually contain several processing steps, it is difficult to define the input resources with all the information required to control each one of them.



- **Duplication:** the processes that have to be performed specifically to generate and print the proof are, in nature, the same processes that are performed to generate and print the final data that has to be printed on the press. This means that very similar information has to be used to define the different processes and, in some cases, the resources used in each case were different, causing duplication in the definition of resources.

As an example of these limitations in JDF 1.1, the color management stage for proofing was defined using some of the *ProofingParams* subelements, but not all the information that is required to control a full color management stage was included. However, this stage equivalent to a color management transformation, defined in JDF 1.1 with *ColorSpaceConversion* process, which uses a different resource to define the transformation parameters.

The solution taken to address these issues in JDF 1.2 is to deprecate the *Proofing* and *SoftProofing* processes and use, instead a combined process to specify the proofing workflow. By doing so, the job ticket explicitly defines the processing that has to be performed and provides the flexibility to implement proofing solutions that can be easily integrated in different workflows.

Once it was decided to deprecate the *Proofing* process, certain changes had to be done to ensure that the atomic processes that were going to be combined to define the proofing specific processing, had all the information required to be able to specify the different configurations and options.

Some of the things that had to be added and changed are:

- Ability to define how spot colors and DeviceN color spaces are transformed in a *ColorSpaceConversion* process. This implied adding new values to the *SourceCS* enumeration and adding additional attributes to specify spot colors and DeviceN color spaces in the *ColorSpaceConversionOp* resource.
- Settings to control the printing of a proof in the *ImageSetting* process.

These changes are described in more detail in the following sections.

3. Combined Processes for Proofing

Due to the diversity of proofing workflows and applications, it is not possible to come up with a unique way of describing proofing with combined processes. The actual processes that have to be combined will depend on the specific capabilities of the RIPs and devices used for proofing and the whole production workflow in which the proofing is done. However, in order to start describing how to map the deprecated *Proofing* process into a combined process, we will define a “generic” combined process that may be used to describe the proofing step in a workflow.

The generic combined proofing process combines the following JDF processes:

- *ColorSpaceConversion (1)*: this process converts the contents of the input *RunList* from the several different input color spaces to the color model (process color space plus supported spot colors) of the press (production device).
- *Interpreting*: interprets the input *RunList* file(s) and converts them to an internal display list suitable to be processed by the *Rendering* process.
- *Rendering*: renders the raster data.
- *ColorSpaceConversion (2)*: converts the data from the press color model to the proofer device color model.
- *Screening*: screens the raster data.

- *Imposition*: combines the pages and marks on the imposed sheets. This process is only required if imposition proofing is done.
- *ImageSetting* or *DigitalPrinting*: at the end of the combined process, there must be a process that specifies the actual printing of the proof. Depending on the characteristics of the proofing device, *ImageSetting* or *DigitalImaging* can be used. The parameters (resources) for each case are different.

A note regarding the ordering of atomic processes in the combined process: there are several possible orders that may achieve the same result; for example, the two color space conversions could be chained at the beginning of the sequence or they could be done after rasterization. However, there are certain precedences that must be respected: the first color space conversion must be done before the second one, rasterization must be done after interpreting, screening must be done after rasterization and the second color conversion, and the printing of the proof (*ImageSetting* or *DigitalPrinting*) must be done after screening. The capability of processing equivalent combined nodes with different nodes will depend on the particular device executing the process.

The following figure (Fig 1) shows a process network in which a combined process similar to the one described is used to define the process of printing a page proof. In this case, the proof is generated directly out of the input file, thus, this workflow depicts a “RIP twice” proofing solution, where the input file(s) are processed twice: once for proofing and once for rendering the data to be image set.

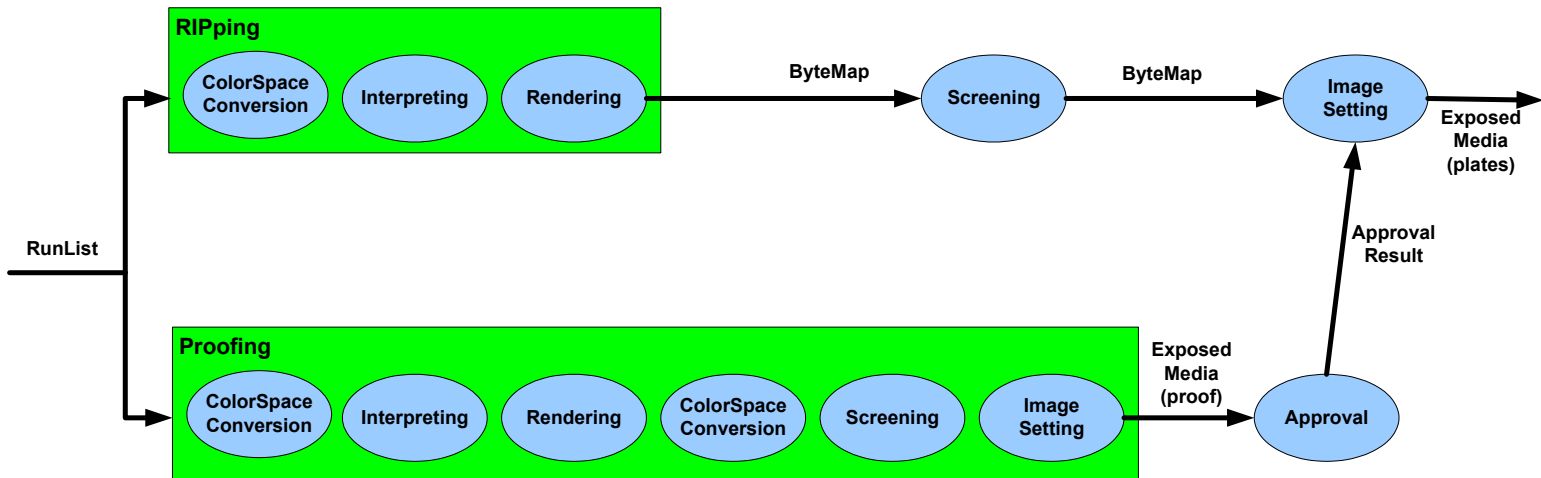


Fig 1 Process network describing a workflow with proofing and approval.

There are many other combinations of JDF processes that may be used to describe the printing of a proof, depending on the specific workflow used. As an example, the following figure shows a different process network to describe a different workflow, but with the same result than the previous one. In this case, the input file is processed (RIPed) once in the RIPping process and then the raster data is output to a TIFF file. The TIFF file is used as input to the proofer device, which only has to do one color conversion (from the press color model to the proofer color model). Thus, this process network describes a “RIP Once, Output Many” (ROOM) workflow.

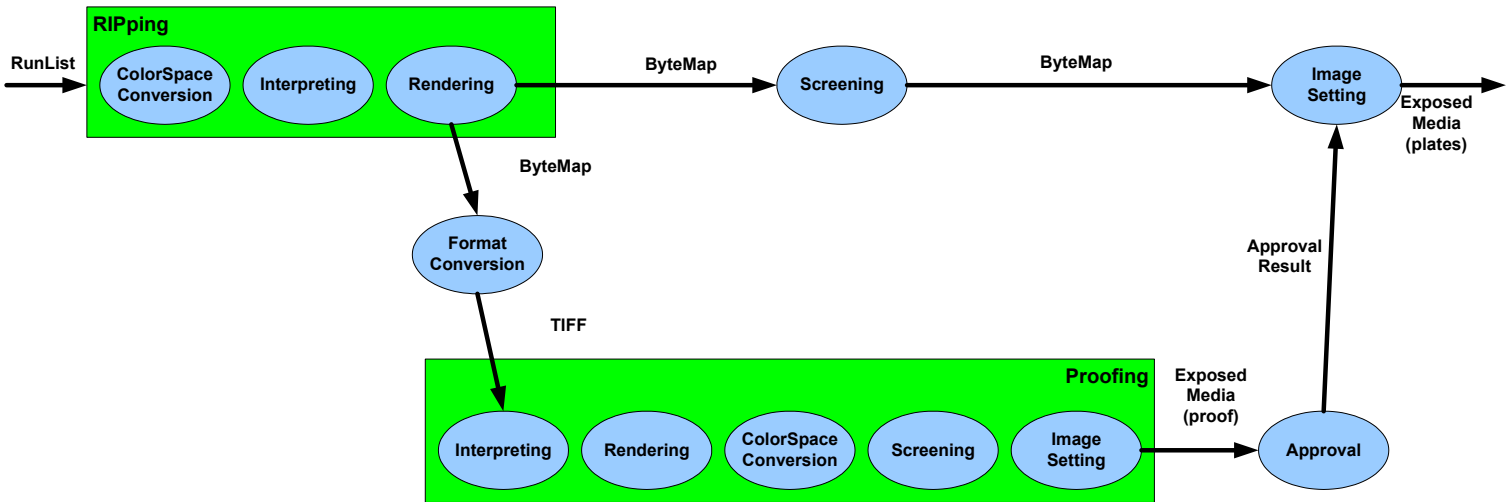


Fig 2 Process network describing a ROOM (RIP Once Output Many) workflow with proofing and approval.

4. Combined Processes for SoftProofing

In a similar way than in the previous section, we can define a “reference” soft proofing combined process. The main difference between (hard) proofing and soft proofing is that in the later, no printed proof is produced, so no *ImageSetting* or *DigitalProofing* processes are combined. Instead, the rasterized data (the bytemap) is sent directly to the *Approval* process. This process must implement the user interface to show the user the rasterized data on the display and allow her to annotate the proof and approve/reject it; possibly using digital signature technology. Fig 3 shows a process network describing a workflow that uses soft proofing.

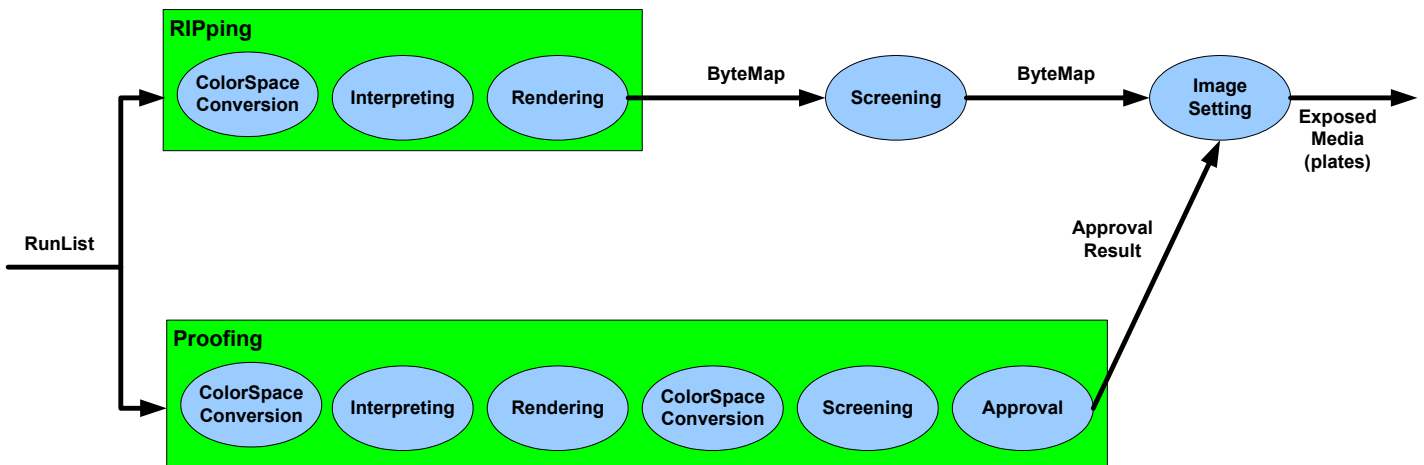


Fig 3 Process network describing a workflow with soft proofing.

The same considerations about the ordering of the atomic process in the combined process described in the previous section for the proofing combined process apply to soft proofing.

5. Considerations on JDF Atomic Processes

As explained in the previous sections, the exact JDF atomic processes that will be used to describe proofing and soft proofing will depend on the specific workflow to implement. However, there are some JDF “atomic” processes that will be commonly used. This section analyzes some of the issues to consider when using these processes in proofing workflows.

5.1. ColorSpaceConversion

In a production workflow with proofing, there must be two color space conversions:

- The "**input color conversion**": converts the color spaces in the input assets to the production device (press) color space.
- The "**proofing color conversion**": converts the color spaces from the production device color space to the proofer device color space.

In JDF 1.2 these two conversions must be specified with two different *ColorSpaceConversion* processes. Whether the two processes are included in the same combined process or not will depend on the exact workflow being described and the capabilities of the different devices involved. As an example, Fig 1 and Fig 2 show two different workflows: in the first case, the two *ColorSpaceConversion* processes are combined in the proofing combined node, but in the second example, they are executed in two different combined nodes (the ripping and the proofing nodes).

5.2. Interpreting and Rendering

In most of the cases, the input data to the proofing combined process will be in a format that requires interpreting and rendering; so these two processes will usually be included in the combined process describing the proofing step in the workflow. The only format that does not require interpretation is the JDF *ByteMap*.

5.3. Screening

There are two possible cases to handle screening:

- If the proofer device can emulate the screening of the press device (i.e. it is a "screen proofer"), the *Screening* process should be performed once at the ripping combined process and the halftoned data should be sent directly to the proofing combined process.
- If the proofer device is a "contone proofer" (i.e. it does not have the capability of emulating the screening of the press device), there must be two *Screening* processes: one for the press device and one for the proofer device. The workflows shown in Fig 1, Fig 2 and Fig 3 are all of them cases of contone proofing (i.e. different screening for press and proofer).

5.4. ImageSetting/Digital Printing

In order to actually print the proof(s) either an *ImageSetting* or *DigitalPrinting* process has to be specified at the “end” of the proofing combined process. This process defines how the proof is actually printed.



5.5. Approval

The approval process must be executed to get the user (customer) approval before the final production printing can be started. The implementation of the Approval process will require a user interface

6. Mapping of Deprecated *Proofing* and *SoftProofing* Resources

This section maps the resources of the deprecated Proofing and SoftProofing processes to the resources of a generic combined proofing and soft proofing processes described in sections 3 and 4.

Proofing/SoftProofing resource	Combined Process		
	Resource	Atomic Process	Comments
<i>ColorantControl</i>	<i>ColorantControl</i>	<i>ColorSpaceConversion(1)</i>	The <i>ColorantControl</i> resource in the deprecated <i>Proofing</i> and <i>SoftProofing</i> processes describes the color model of the job as going to be printed on the production device (press). Consequently, in the combined process the resource must be applied to the first (input) color transformation.
<i>ColorSpaceConversionParams</i>	<i>ColorSpaceConversionParams</i>	<i>ColorSpaceConversion(1)</i>	The <i>ColorSpaceConversionParams</i> resource in the deprecated <i>Proofing</i> and <i>SoftProofing</i> processes describe the transformation from input space to press space. Consequently, in the combined process the resource must be applied to the first (input) color transformation.
<i>Media</i>	<i>Media</i>	<i>Interpreting, Rendering, Screening, ImageSetting, DigitalPrinting</i>	All these processes have as input a <i>Media</i> resource that describes the media on which the proof is going to be printed; so the <i>Media</i> resource must be an input to any of them used in the combined process chain.
<i>RunList</i> (Document)	<i>RunList</i>	First atomic process in the combined process	This is the input data to the proofing combined process; so it has to be an input to the first process in the combined chain that consumes the input data.
<i>Layout</i>	<i>Layout</i>	<i>Imposition</i>	These resources are only used in the case of imposition proofing.
<i>RunList</i> (Marks)	<i>RunList</i> (Marks)	<i>Imposition</i>	



Proofing/SoftProofing resource	Combined Process		
	Resource	Atomic Process	Comments
<i>ProofingParams/@ColorType</i>	<i>ColorSpaceConversionParams/FileSpec</i> (FinalTargetDevice)	<i>ColorSpaceConversion(2)</i>	The target profile specified in the second color transformation (proofing transformation) controls whether the final proof will be monochrome or color.
<i>ProofingParams/@DisplayTraps</i>		<i>Trapping</i>	Whether traps are displayed or not in the proof will depend on two conditions: - the input to the combine proofing process has already been trapped - there is a <i>Trapping</i> process included in the combined process.
<i>ProofingParams/@HalfTone</i>	<i>ScreeningParams</i>	<i>Screening</i>	The screening parameters used in the <i>Screening</i> process determine whether the same screening used in the final press is used for proofing.
<i>ProofingParams/@ImageViewingStrategy</i>	<i>ImageReplacement/@ImageReplacementStrategy</i>	<i>ImageReplacement</i>	If the strategy desired for image viewing is different than <i>OmitReference</i> , an <i>ImageReplacement</i> process must be inserted in the combined process to control the replacement of images embedded in the input file by lower resolution versions or proxies.
<i>ProofingParams/@ProofingRenderingIntent</i>	<i>ColorSpaceConversionParams/ColorSpaceConversionOp/@RenderingIntent</i>	<i>ColorSpaceConversion(2)</i>	The rendering intent specified in the second color transformation (proofing transformation) controls the rendering intent in the proof.
<i>ProofingParams/@ProofType</i>		<i>Imposition</i>	Whether the proof type is page or imposition will depend on whether an <i>Imposition</i> process is included in the combined process or not.
<i>ProofingParams/@Resolution</i>	<i>RenderingParams/ObjectResolution</i>	<i>Rendering</i>	The <i>ObjectResolution</i> element(s) in the <i>RenderingParams</i> resource define the resolution at which the data is rendered for proofing.
<i>ProofingParams/FileSpec</i>	<i>ColorSpaceConversionParams/FileSpec</i> (FinalTargetDevice)		The proofer profile is the output profile of the second color transformation
<i>ProofingParams/MediaSource</i>	<i>Media</i>	<i>Interpreting, Rendering, Screening, ImageSetting, DigitalPrinting</i>	All these processes have as input a <i>Media</i> resource that describes the media on which the proof is going to be printed; so the <i>Media</i> resource must be an input to any of them used in the combined process chain.