TIFF Technical Note 1: TIFF Trees

Motivation

TIFF has always supported what amounts to a singly linked list of IFD's in a single TIFF file, via the "next IFD pointer," though most applications currently ignore any IFD beyond the first one.

Probably the best use for a linked list of IFD's is when you want to store multiple different but related images in the same file. The classic example is multiple pages of a fax transmission.

But suppose we want to define an image mask for an image. (This is defined within the TIFF spec, but currently rarely used.) That would mean two IFD's in the file, one for the main image and one for the image mask. Perhaps the image is really large, so that we would like to include a lower resolution version (or "subimage") of the image; if so, we must include a lower resolution version of the image mask, too. Now we have four IFD's in our TIFF file. The order of the files in the linked list is nowhere defined in the TIFF spec, so the TIFF reader has to be careful that it doesn't try to use the wrong subimage for the wrong purpose. The situation is still workable, but is starting to get complicated.

Let's complicate the scenario further by saying that we want to include subimages and masks for each of the pages in a multiple-page fax transmission. Now things are really messy. Which subimage goes with which main image? We could make a list of rules, but they would be quite arbitrary and difficult to use.

Solution

If only we had the concept of a tree within a TIFF file, we would have a natural way to associate a main or "parent" image with a subordinate or "child" image such as a reduced resolution version or an image mask.

One way to create a tree structure within a TIFF file is to define a new **SubIFDs** tag, of type LONG. Each LONG value points to a "child" IFD structure.

This method is safe because old TIFF readers will not recognize the new tag, and will simply ignore it. It is versatile because there is no built-in restriction concerning what you store in a child IFD—a feature that may come in handy for future private and public enhancements.

So when should you use the NextIFD pointer at the end of an IFD and when should you use the **SubIFDs** tag? Use the NextIFD pointer if your application requires that multiple visually unrelated images be stored in the same TIFF file—a multi-page fax transmission, for example. Use the **SubIFDs** tag for pointing to images that modify or add information to or otherwise "help" the Parent image—transparency masks and subsampled versions, for example.

No particular order or precedence is defined for child images. For example, if we have two child images, such as a transparency mask and a subsampled version, they can appear in either order.

New Tag

SubIFDs
Tag = 330 (14A)
Type = LONG
N = number of child IFDs

Each LONG value is an offset (from the beginning of the TIFF file, as always) to a child IFD. Child images provide extra information for the parent image—such as a transparency mask or subsampled version of the parent image.

Examples

Let's revisit our first example. We have a main image, a mask for the main image, a low res subimage, and a mask for the low res subimage. The IFDs in our file would look like this:

```
              Main Image
                   |
          ________________________
          |                      |
    LowRes Child            Mask Child
          |
  LowRes Mask Child
```

The **SubIFDs** tag is used in both the main image and the LowRes Child image. In the main image, the **SubIFDs** tag has two values, which point to the beginning of the IFD structures for a LowRes Child and Mask Child images. In the LowRes Child image, the **SubIFDs** tag has only one value, the location of the IFD for a LowRes mask image.

If there is more than 1 child image for a given parent image, the NextIFD value of Child #1 must point to Child #2, and so on. The last Child's NextIFD value must be zero.

This completes the TIFF Tree tech note.