

Architectural Design of Metadata for Images

James R. Milch and George E. Sotak Jr.
Eastman Kodak Company
Rochester, New York, USA

Abstract

Digital imaging on the Internet and on personal computers has evolved into a very open system. Most imaging workflows on the desktop involve hardware and software from several different companies. The creation of industry standards has resulted in a reduction in the number of image file formats that users encounter. Device-to-device color variation has been reduced by the introduction of sRGB as an exchange color space. Still, desktop digital imaging is difficult to use. It is now widely recognized that the user experience can be improved by carrying non-imaging information ~~im~~age metadata ~~al~~ong with the pixel values. However, existing infrastructure for image metadata diminishes its utility. Many of the standard image file formats support the storage of metadata, but their storage mechanisms are simplistic and impose unnecessary constraints on the metadata representation.

This paper examines the use of metadata in the most common consumer and business workflows. From these use cases, we extract the key requirements for the representation of the metadata and the subsystems that handle it. These cover issues such as the metadata carrier, the metadata's representation and semantics, metadata retention, and metadata management. We then explain the basic architecture of a system that meets these requirements. This architecture is supported by a comparison between this domain and other, better studied domains, such as information exchange across heterogeneous databases and the design of the World Wide Web. Our architecture does not answer all the issues raised by the requirements, but it provides a context in which to address them.

Introduction

The growth of digital photography is driven by three technological forces: the development of inexpensive digital cameras, the availability of inexpensive computing power, and the growth of ubiquitous networking. If the first force were the dominant one, digital photography would be simple. Instead of capturing pictures with film and getting optical prints, consumers would be capturing pictures with CCD imagers and getting digital prints. That is not the case. The real demand for digital photography comes from the new things that consumers can do with digital images, no matter how they were captured.

Digital images are information. They can be integrated into information systems for storage, searching, transmission, or analysis. Of course, in the end, they must be viewed in soft copy or hard copy form to be enjoyed. However, the opportunities for enjoyment can be greatly increased by those information system functions.

The open systems we use today to capture, manage, and use digital images have an important shortcoming. They provide means to work with the image, but no means to manage data about the image. This additional information, called metadata, may describe objects in the scene, the state of the camera at the moment of scene capture, the desires of the photographer, or the purpose of the image. The metadata may be used to improve the image quality of the reproduction, help photographers to organize their image collections, or simplify the overall picture workflow.

Several common image file formats support metadata. For example, TIFF files and Exif files may contain collections of tags that contain information about the image. However, these formats and the tags defined for them were designed to be used either in closed systems or in open systems with constant human intervention. As we look at using them in modern information systems, they have two clear shortcomings. First, the data are organized in a rigid and cumbersome manner. It is difficult to audit the tags in a file for validity or add new tags on the fly. Second, the meaning, usage, and interrelationships of the tags have not been carefully specified. Every product uses the tags just a little differently. This makes it very difficult to use the tags reliably in automated processes, as part of an open system.

As we looked more closely at the design of an architecture for photographic metadata, we realized that it was not just a "plumbing problem". There are some very difficult issues to face. We also found that the problem is closely linked with several other active research areas. In particular, we found parallels in:

- Work done by the computer science community to improve the design of the World Wide Web
- Work done by the bibliographic community to index and organize objects in collections of many kinds
- Efforts to define the exchange of information and intent between "intelligent agents".

This paper will draw on experience in photography and observation from these other domains to define a basic

architecture for photographic metadata and the software that handles it. This architecture supports use of the metadata both in a purely photographic environment and as part of the larger information web. Some of this work was done in support of a Digital Imaging Group (DIG) working team that has been examining the use of metadata in photography [1].

Metadata Requirements

The first step in understanding the requirements for describing and handling metadata is to collect a series of customer use cases. Each use case describes, in narrative fashion, one particular photographic experience. It is written from the customer's point of view, but includes interactions with equipment, software, and service providers. We have written use cases that describe the photographic needs of consumers and casual business users. We did not study the needs of professional photographers or publishers. Because of space limitations, we have omitted example use cases from this paper.

From these use cases, we have derived a set of photographic metadata requirements. These requirements apply not only to the metadata itself, but also to the software layers that are responsible for managing the metadata. The technology to satisfy some of the requirements comes from Computer Science; other requirements deal more with Imaging Science issues. We cannot list all of the requirements here, but the must critical ones are presented. The ** notation is explained in the following section.

Requirements on the representation of the metadata

- Provide interoperability across hardware platforms, software environments, and usage domains **
- Provide ability to add new metadata items without a central registration authority **
- Support validation of the syntax and content of the metadata
- Support hierarchical grouping of, and meaningful relationships between, metadata items

Requirements on the means used to store a collection of metadata in an image file with pixel data

- Support the definition of blocks of metadata
- Support compression, encryption, and priority selection of metadata by block
- Support rapid access to specific blocks of metadata

Requirements on the software that manages the metadata

- Provide interfaces that are independent of storage format, recognizing that there will be more than one
- Create an effective in-memory representation of the metadata with methods to use it
- Implement support for the metadata representation and storage features noted above
- Provide the ability for an application to copy all the metadata from one file to another, based on the syntax of the metadata.
- Provide means to automatically update the metadata in response to changes made in the image, keeping the two of them synchronized. **
- Extract useful metadata about the scene and the quality of the image from the image data itself. **

Key challenges

Some of the requirements given above can be implemented using careful design and judicious selection of known technologies. Others present real scientific challenges and cannot be completely satisfied by current technology. We believe that the four requirements marked above with "**" should be the subject of further research.

Lessons from Other Fields of Research

If the metadata associated with images is used only as an occasional hint for a human user, the precise way in which it is defined and stored is not very important. However, the real potential for making imaging easier with metadata depends on software that can use metadata to make "intelligent" decisions about the content or usage of images. Furthermore, today's business models require a dynamic extensible, architecture. These two facts call for careful consideration of the definition, storage, and syntax of metadata.

The problem of metadata architecture for pictures is closely linked with several other active research areas. Libraries, museums, and other collections store a wide variety of artifacts, including books, pictures, maps, and fossils. The bibliographic community is facing the challenge of converting the "card catalog" information on these objects into a universally-searchable, interoperable form [2]. From their point of view, pictures are another kind of artifact. This research community has made particular contributions to understanding how to describe the relationships between objects [3].

The term "metadata" has been applied in the database-design community to describe information somewhat different than our image metadata. Here, metadata describes the database structures created to store, define, and relate items in a database. This metadata becomes important if one wishes to move information between domains described by two different databases. Jim Fulton has called this challenge 'Semantic Plug and Play" [4]. This technical community has stressed the importance of formal data models as key tools.

Perhaps the most informative lessons for designing image metadata come from the World Wide Web. The hallmarks of the WWW are flexibility, extensibility, and distributed development. Tim Berners-Lee has described the design philosophy of the World Wide Web in a series of "Design Issues" [5]. We repeat below a few of his key concepts and connect them to image metadata.

The principle of least power

There are classes of computer languages that vary in their ability to express complex structures and behavior. They range from purely descriptive (e.g., HTML), through declarative languages such as XML, to procedural languages such as Java. In selecting the language in which to define the metadata, choose the language with the least power that can meet the needs. This may make a current application that uses the metadata more complex (to make up for the limitations of the language), but it makes all the future applications, unknown to the present creators of this metadata language, simpler and cleaner. It is better to leave the complexity to a separate, more easily changed layer of the system. We have followed this advice and chosen XML, a declarative language, to describe image metadata.

Decentralization

Do not allow a distributed system to depend upon a central control point or registration authority. This produces fragile and brittle applications. This principle is expressed in the requirement for adding new metadata and discovering previously unknown metadata on the fly.

Test of independent invention

"If someone else had already invented your system, would theirs work with yours?" Use this simple thought experiment to assure that your system could be part of an as-yet unspecified larger system. For example, the metadata architecture for images should fit easily within an architecture for a wider class of objects, even if that architecture has not yet been defined. Our architecture supports late binding of meaning with syntax, to assure the flexibility suggested by this principle.

Tolerance

"Be liberal in what you require but conservative in what you do." The component that uses image metadata should be forgiving about variations in the metadata it receives, as long as it can unambiguously act correctly. This principle reflects a move away from hardware implementations, for which liberality means great complexity, to software implementations of metadata management.

Trust

Each item of metadata makes an assertion about some aspect of an image. To what degree can the receiver of the metadata trust this assertion to be true? This aspect of metadata is particularly important when intelligent agent software is using the metadata to act in the user's name. Our present ability to associate authority and accuracy with metadata items is still primitive. The architecture described here includes the notion of delivering "more detail" when asked, and the ability to encrypt critical metadata.

Metadata Architecture

A metadata architecture must describe two key elements: the means of expressing a collection of metadata and the functionality required in the software that manages the metadata. This is shown in the context of an imaging application in Fig. 1. The application relies on a variety of "middleware" components, or toolkits, each of which performs a specific function well. Key imaging components shown in Fig. 1 are the Image Processing Toolkit, the Metadata Manager, and an Image Analysis Package. The Image Processing Toolkit and the Metadata Manager are responsible for interfacing to the image file, which contains both pixel data and metadata. Just as modern applications rely on an Image Processing Toolkit to fetch pixels from the file and perform elemental image processing operations, they should expect the Metadata Manager to fetch metadata by name and perform routine metadata operations. Another significant component shown in Fig. 1 is the Image Analysis Package. This is a collection of routines that use Image Understanding algorithms to derive meaningful metadata from the pixel values [6].

The representation of the metadata and the functionality of the Metadata Manager may be organized into three levels of abstraction: syntax, semantics, and context.

The syntax level addresses the structural representation of the metadata. For the receiver of the metadata, it answers the question, "How do I group these bytes or characters into meaningful components?" Semantics concerns the conveyance of the meaning of the metadata. It answers questions like, "What assertion is intended by this sequence of components?" Context addresses the contextual usage of the metadata. It is based on domain and experiential knowledge, and it answers questions like, "Based on these assertions, what is the appropriate image processing for this image?" There is a clear dependence between these levels. The translations of the context level rely on the well-formed meanings in the semantic layer. The semantic layer, in turn, relies on the syntax layer for the well-formed representations.

The ideal image metadata system would allow a device near the beginning of the image chain to add a new metadata item to the image and communicate at all three levels with a device at the end of the image chain. This would enable new end-to-end features to be introduced into open systems without end-to-end software revisions. We are not yet able to describe an architecture that makes this possible.

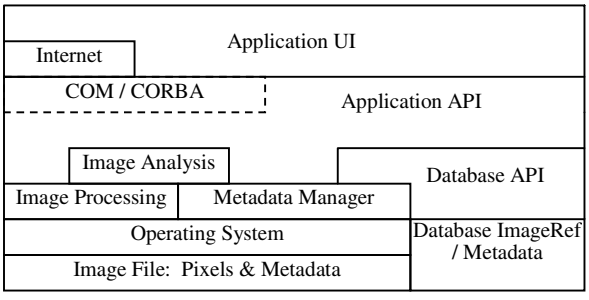

Figure 1. Application architecture

However, the syntax level forms the foundation upon which all other functionality will be built. This is also the level that can be addressed with today's technology and, therefore, will be primary topic of the remainder of this paper. We will touch upon the semantic and context layers to convey the particular challenges that they pose.

Syntax

The syntax level addresses the representation of the metadata. One common approach is a loosely organized collection of tag-value pairs. This is the method used in the TIFF format [7]. The tag is a numeric value assigned by a central TIFF tag registration authority. The only attribute of a value is its type, limited to a specified set of primitive data types such as byte, ASCII, and short. This syntax fails to meet many of the requirements stated above and forms a poor foundation for the powerful layers we would like to build on top of it.

The World Wide Web Consortium has created a modern declarative language to express the structure of WWW documents. This language, XML (eXtensible Markup Language) [8], has many of the characteristics we seek for image metadata. This is no surprise, given the commonality of system requirements. XML is also being used by others as a means to define common data structures for passing information between heterogeneous information systems. In addition, XML fulfills our desire to enable digital images to participate in resource discovery on the World Wide Web.

We will not describe the details of XML here. Several good references are available [9, 10]. Here is an example of metadata expressed in XML:

```
<Person>
    <FirstName> John </FirstName>
    <LastName> Doe </LastName>
    <Birthdate> 1-1-1934 </Birthdate>
</Person>
```

XML consists largely of nested sets of "elements", delimited by angled brackets that identify the nature of the content they surround.

We propose the direct use of XML text to store metadata. This will simplify distributed development of imaging applications and establish a firm foundation for powerful semantic and contextual capabilities. At first glance, XML appears to be grossly wasteful of storage space, compared to binary tag-value pairs. It will take up somewhat more space (TIFF structures are themselves rather inefficient), but we have addressed this issue in several ways. First, the element names are intended to be human-readable, but some restraint should be used in giving elements fully expressive names. Second, large binary structures, such as ICC profiles, will still be stored as binary blobs. Third, standard data compression methods can be used to compress the text if metadata size becomes an issue.

The XML language supports flexible grouping of metadata. This capability should be used to organize related metadata items in order of importance. For example, the top level of geographic location information might be latitude and longitude. Beneath that, data might be available on the source of the information (GPS, cell phone network), further details (satellite id), and its reliability. Of course, any organizational choices at this level impose current opinions of "importance" on the data structures. However, no data is lost in this process, only hidden.

One important capability of the XML language is validation. It is possible to encapsulate the syntax allowed for all image metadata into a compact "Document Type Declaration", or DTD. The DTD is not required to understand the metadata, but an XML parser can unambiguously validate any example of metadata against the DTD. This clearly improves the accuracy of communication between sender and receiver.

As presently defined, XML does not meet all of our requirements for image metadata. It is able to validate the form of a metadata example, but not its content. As a simple example, there is no means to check that a numerical month falls between one and twelve. The World Wide Web Consortium (W3C) recognized this shortfall and launched the XML Schema Working Group in early 1999 to define a full-featured declarative language based on XML [11]. At the time of this writing, XML Schema is still a W3C Working Draft, but it appears to contain the capabilities we seek for image metadata. For example, it permits the definition of the type "date" with a specific substructure and permitted values.

Semantics

The meaning of the metadata is not always obvious from its syntax. The human-readable names in XML help, particularly if a human is present to read it. For example, XML Stylesheets [12], already supported by some browsers, can be used to present metadata to a user with no custom programming.

Software applications or agents have harder task. There is substantial debate in the broader metadata community over the ability of the basic XML syntax to communicate meaning. An alternative language, "Resource Description Framework", or RDF, has been proposed [13]. RDF is defined in terms of XML syntax, but has the additional capability to express relationships as directed labeled graphs. For example, we may wish to communicate, "In this picture, John is standing behind Mary." There is no natural way to express this in the basic XML syntax so that the meaning is unambiguous. RDF is designed for this type of assertion. Our own studies indicate that XML will suffice for metadata that is closely associated with a single image and describes that entire image. Metadata about some sub-element of the image, or metadata that attempts to express the relationship between two elements may require the more powerful syntax of RDF.

Context

We have no general solution to the context layer for metadata today. Each application or software agent must

contain explicit code for every action it takes in response to the metadata in the image. Several of the key challenges noted above fall into the context layer.

Metadata manager implementation

The realization of this architecture is encompassed in the Metadata Manager component of Fig. 1. The current implementation focuses on providing storage format-independent access to metadata. The syntax level of the architecture plays a key role in this, providing the common ground to which all other metadata storage representations are mapped.

We have developed a generic object-oriented class model for the in-memory representation of the XML-based metadata representations. All programmatic-based manipulation of metadata uses this class model. The XML-based metadata definitions are parsed to build a class factory of prototype instances. Applications and other components of the Metadata Manager use this factory to instantiate valid metadata instances.

The storage format input/output variations have been abstracted out to a common class interface, referred to as Accessors. The Accessors have explicit understanding of the file format and common metadata representations and, therefore, are able to perform the mapping between the two representations. The Accessors' interface uses the metadata class model, completely hiding any file format details from the application. The Accessors also provide special read / write all metadata methods. These are present to fulfill the requirement to "copy all metadata". Presently, the Accessors support the Exif, TIFF, and APS MOF (Magnetics On Film) file formats.

Conclusion

We have described the requirements for a general architecture for creating, storing, and using metadata with images. The elements of the syntax layer are clearest; the semantic and context layers are largely open for future research. The Digital Imaging Group (DIG) working group on image metadata has recommended to the JPEG2000 working group [14] that the JPEG2000 file format use XML as the syntax for flexible metadata storage. This same group has defined the container structures that will be used to combine image data and metadata into a single file. We support this choice. As soon as XML Schema becomes a standard, we expect to enhance the capabilities of JPEG2000 metadata management with schemas. This will close some existing weaknesses in the XML standard.

We believe that XML establishes a firm foundation for capabilities based on semantics and context. Some of these challenges are common to broader uses of metadata for bibliography and the World Wide Web. Others are peculiar to images and will be solved largely within the field of Imaging Science. In spite of the many problems left to be solved, the industry is moving forward to implement basic solutions now, in a manner that provides a good foundation for future implementations.

References

1. *The Power of Metadata Is Propelling Digital Imaging Beyond the Limitations of Conventional Photography,* Digital Imaging Group, http://www.digitalimaging.org/i_dig35_form.html
2. Dublin Core Metadata, http://purl.oclc.org/dc
3. *Guidance on expressing the Dublin Core within the Resource Description Framework*, http://www.ukoln.ac.uk/metadata/resources/dc/datamodel/WD-dc-rdf/
4. Jim Fulton, *Semantic plug and play*, prepared for the Joint Workshop on Standards for the Use of Models that Define the Data and Processes of Information Systems, September, 1996, Seattle, Wa. Available at http://www.mel.nist.gov/workshop/jtc1-96/papfultn.htm
5. Tim Berners-Lee, *Axioms of Web Architecture*, September, 1999, http://www.w3c.org/DesignIssues/Principles.html
6. Gupta, A., Ramesh, J., Visual Information Retrieval, *Communications of the ACM*, Vol. 40, No. 5, 1997.
7. TIFF (Tagged Image File Format) Revision 6.0, June, 1992. The TIFF specification is maintained by Adobe Systems at http://partners.adobe.com/asn/developer/PDFS/TN/TIFF6.pdf
8. eXtensible Markup Language, http://www.w3.org/XML/
9. Walsh, N., *A Technical Introduction to XML*, http://www.xml.com/pub/98/10/guide0.html
10. J. Bosak and T. Bray, XML and the second-generation web, *Scientific American*, May 1999
11. XML Schema Structures and Datatypes Working Drafts, http://www.w3.org/TR/xmlschema-1, http://www.w3.org/TR/xmlschema-2
12. eXtensible Stylesheet Language (XSL) http://www.w3.org/Style/XSL/
13. Resource Description Framework http://www.w3.org/RDF/
14. JPEG2000 Working Group, http://www.jpeg.org/public/jpeglinks.htm

Biography

James R. Milch is Manager and Chief Architect of the Image Data Systems Program at Eastman Kodak Company. He is responsible for the integration of Kodak products so that they deliver end-to-end benefits to customers in an open-systems environment. His past assignments at Kodak ranged from conceptual research to the development of specific products. Dr. Milch earned a B.S. in Physics from Yale University and a Ph.D. in Physics from Princeton University.

George E. Sotak Jr. is a Research Associate in the Image Science Technology Laboratory at Eastman Kodak Company. He is currently the Project Leader of the Metadata Manipulation Architecture project in the Image Data Systems Program. His past assignments at Kodak include research and development of Visual Information Management technologies and algorithm and system development for the application of image processing and analysis to all disciplines of microscopy. Mr. Sotak earned a B.S. in Electrical Engineering and Applied Physics from Case Western Reserve University and a M.S. in Electrical Engineering from The Ohio State University.