Technician Education in Additive Manufacturing & Materials
TEAMM
4TEAMM.org

Standard Specification for Additive Manufacturing File Format (AMF)

Gary Coykendall, Author

Abstract
This module provides: an overview of the standard specification for additive manufacturing file format.

Objective:
The student will be able to:
Define the additive manufacturing file format.
Explain the various terminologies used the additive manufacturing file format
Contrast and compare the similarities and differences between the .stl and the .amf file formats.

AM core competencies addressed (most important in bold)
9Aa1 Define Additive manufacturing file format (.amf)
9Aa2 Compare and contrast the .amf format to the .stl format
9Aa3 Explain and illustrate the advantages of using the .amf format

Key Words: .AMF file format, .STL file format, extensible markup language XML

Type of Module: Discussion

Time Required: 1 hour

Grade Level
Grades 8 through college

Equipment and supplies needed
Standard classroom set-up (computer, overhead projection)

Additive Manufacturing File Formats:

A. Please refer to the accompanying PowerPoint presentation titled: Standard Specification for Additive Manufacturing File Format (.AMF)

B. .STL File Format (see slide 2)

a. STL file format has been the industry standard for transferring information between design programs and additive manufacturing equipment.

b. Contains information only about a surface mesh and has no provisions for representing color, texture, material, substructure, and other properties of the fabricated target object.
c. Additive manufacturing technology is quickly evolving from producing primarily single-material, homogenous shapes to producing multimaterial-geometries in full color with functionally graded materials and microstructures.
d. There is a growing need for a standard interchange file format that can support these features.

B. Key Considerations (see slide 3)

a. Technology Independence

1. File format shall describe an object in a general way such that any machine can build it to the best of its ability.
2. It is resolution and layer thickness independent and does not contain information specific to any one manufacturing process or technique.

b. Simplicity

1. The AMF file format is easy to implement and understand.
2. The format can be read and debugged in a simple ASCII text viewer to encourage understanding and adoption.
3. No identical information is stored in multiple places.

c. Scalability

1. The file format scales well with increase in part complexity and size and with the improving resolution and accuracy of manufacturing equipment.
2. This includes being able to handle large arrays of identical objects, complex repeated internal features (for example, meshes), smooth curved surfaces with fine printing resolution, and multiple components arranged in an optimal packing for printing.

d. Performance

1. The file format should enable reasonable duration (interactive time) for read-and-write operations and reasonable file sizes for a typical large object.

e. Backward Compatibility

1. Any existing STL file can be converted directly into a valid AMF file without any loss of information and without requiring any additional information.
2. AMF files are also easily converted back to STL for use on legacy systems, although advanced features will be lost.
3. AMF files are also easily converted back to STL for use on legacy systems, although advanced features will be lost. This format maintains the triangle-mesh geometry representation to take advantage of existing optimized slicing algorithm and code infrastructure already in existence.

f. Future Compatibility

1. To remain useful in a rapidly changing industry, this file format is easily extensible while remaining compatible with earlier versions and technologies.
2. This allows new features to be added as advances in technology warrant, while still working flawlessly for simple homogenous geometries on the oldest hardware.

C. General Structure (see slide 4)

a. Information specified throughout this specification is stored in XML format.

1. The AMF file begins with the XML declaration line specifying the XML version and encoding, for example:

<?xml version=91.09 encoding=9UTF-89?>

2. Blank lines and standard XML comments can be interspersed in the file and will be ignored by any interpreter, for example:

<!-- ignore this comment -->

3. The remainder of the file is enclosed between an opening <amf> element and a closing </amf> element.
4. These elements are necessary to denote the file type, as well as to fulfill the requirement that all XML files have a single-root element. The version of the AMF standard as well as all standard XML namespace declarations can be used, such as the lang(uage) attribute designed to identify the human language used. The unit system can also be specified (mm, inch, ft, meters, or micrometers). In absence of a unit specification, millimeters are assumed.

<amf unit=9millimeter9 version=91.09 xml:lang=9en9>

b. Five Top Level Elements

1. <object>—The object element defines a volume or volumes of material, each of which are associated with a material identification (ID) for printing. At least one object element shall be present in the file. Additional objects are optional.
2. <material>—The optional material element defines one or more materials for printing with an associated material ID. If no material element is included, a single default material is assumed.
3. <texture>—The optional texture element defines one or more images or textures for color or texture mapping each with an associated texture ID.
4. <constellation>—The optional constellation element hierarchically combines objects and other constellations into a relative pattern for printing. If no constellation elements are specified, each object element will be imported with no relative position data. The parsing program can determine the relative positioning of the objects if more than one object is specified in the file.
5. <metadata>—The optional metadata element specifies additional information about the object(s) and elements contained in the file.

D. Geometry Specification

a. The top level <object> element specifies a unique id and contains two child elements: <vertices> and <volume>. The <object> element can optionally specify a

material.

b. The required <vertices> element lists all vertices that are used in this object. Each vertex is implicitly assigned a number in the order in which it was declared starting at zero.The required child element <coordinates> gives the position of the point in three-dimensional (3D) space using the <x>, <y>, and <z> elements.

c. After the vertex information, at least one <volume> element shall be included. Each volume encapsulates a closed volume of the object. Multiple volumes can be specified in a single object. Volumes may share vertices at interfaces but may not have any overlapping volume.

d. Within each volume, the child element <triangle> shall be used to define triangles that tessellate the surface of the volume. Each <triangle> element will list three vertices from the set of indices of the previously defined vertices. The indices of the three vertices of the triangles are specified using the <v1>, <v2>, and <v3> elements. The order of the vertices shall be according to the right-hand rule such that vertices are listed in counter-clockwise order as viewed from the outside. Each triangle is implicitly assigned a number in the order in which it was declared starting at zero (See Slide 5)

e. Smooth Geometry:

1. By default, all triangles are assumed to be flat and all triangle edges are assumed to be straight lines connecting their two vertices. Curved triangles and curved edges can optionally be specified to reduce the number of mesh elements required to describe a curved surface.
2. During read, a curved triangle patch shall be recursively subdivided into four triangles by the parsing program to generate a temporary set of flat triangles at any desired resolution for manufacturing or display. The depth of recursion shall be determined by the parsing program, but a minimal level of four is recommended (that is, convert a single curved triangle into 256 flat triangles).
3. During write, the encoding software shall determine automatically the minimum number of curved triangles required to specify the target geometry to the desired tolerance, assuming that the parser will perform at least four levels of subdivision for any curved triangle.
4. To specify curvature, a vertex can optionally contain a child element <normal> to specify desired surface normal at the location of the vertex. The normal should be unit length and pointing outwards. If this normal is specified, all triangle edges meeting at that vertex should be curved so that they are perpendicular to that normal and in the plane defined by the normal and the original straight edge.
5. When the curvature of a surface at a vertex is undefined (for example, at a cusp, corner, or edge), an <edge> element can be used to specify the curvature of a single nonlinear edge joining two vertices. The curvature is specified using the tangent direction vectors at the beginning and end of that edge. The <edge>

element will take precedence in case of a conflict with the curvature implied by a <normal> element.
6. Normals shall not be specified for vertices referenced only by planar triangles. Edge tangents shall not be specified for linear edges.
7. When interpreting normal and tangents, Hermite interpolation will be used.

f. Restrictions on Geometry (see slide 6)

1. All geometry shall comply with the following restrictions:
    a. Every triangle shall have exactly three different vertices
    b. Triangles may not intersect or overlap except at their common edges or common vertices.
    c. Volumes shall enclose a closed space with nonzero volume.
    d. Volumes may not overlap.
    e. Every vertex shall be referenced by at least three triangles.
    f. Every pair of vertices shall be referenced by zero or two triangles per volume.
    g. No two vertices can have identical coordinates.
    h. The outward direction of triangles that share an edge in the same volume must be consistent. (see slide7)

E. Material Specification (see slide 8)

a. Materials are introduced using the <material> element. Any number of materials may be defined using the <material> element. Each material is assigned a unique id. Geometric volumes are associated with materials by specifying a material id within the <volume> element. Any number of materials may be defined. The material id 909 is reserved for no material (void)

b. Material attributes are contained within each <material>. The element <color> is used to specify the red/ green/blue/alpha (RGBA) appearance of the material. Additional material properties can be specified using the <metadata> element, such as the material name for operational purposes or elastic properties for equipment that can control such properties.

F. Mixed and Graded Materials and Substructures

a. New materials can be defined as compositions of other materials. The element <composite> is used to specify the proportions of the composition as a constant or a formula dependent of the *x*, *y*, and *z* coordinates. A constant mixing proportion will lead to a homogenous material. A coordinate dependent composition can lead to a graded material. More complex coordinate-dependent proportions can lead to nonlinear material gradients as well as periodic and nonperiodic substructure. The proportion formula can also refer to a texture map using the text (texture id,x,y,z) function.

b. Any number of materials can be specified. Any negative material proportion value will be interpreted as a zero proportion. Material proportions shall be normalized to determine actual ratios.

c. Although the <composite> element could theoretically be used to describe the complete geometry of an object as a single function or texture, such use is discouraged. The intended use of the <composite> element is for the description of cellular mesostructures.

G. Porous Materials

a. Reference to materialid 909 (void) can be used to specify porous structures. The proportion of void can be either 0 or 1 only. Any fractional value will be interpreted as 1 (that is, any fractional void will be assumed fully void).

H. Stochastic Materials

a. Reference to the rand(x,y,z) function can be used to specify pseudorandom materials. For example, a composite material could combine two base materials in random proportions in which the exact proportion can depend on the coordinates in various ways. The rand(x,y,z) function produces a random scalar in the range [0,1) that is persistent across function calls.

I. Color Specification

a. Colors are introduced using the <color> element by specifying the RGBA (transparency) values in a specified color space. By default, the color space shall be sRGB but alternative profiles could be specified using the profile attribute. The <color> element can be inserted at the material level to associate a color with a material, the object level to color an entire object, the volume level to color an entire volume, or a vertex level to associate a color with a particular vertex

b. Object color overrides material color specification, a volume color overrides an object color, and vertex colors override volume colors.

J. Graded Colors and Texture Mapping (see slide11)

a. A color can also be specified by referring to a formula that can use a variety of functions including a texture map.

b. When referring to a formula, the <color> element can specify a color that depends on the coordinates such as a graded color or a spotted color. Any mathematical expression that combined the functions described in can be used. For example, use of the rand function can allow for pseudo-random color patterns. The tex function can allow the color to depend on a texture map or image. To specify a full-color graphic,

typically three textures will be needed, one for each color channel. To create a monochrome graphic, typically only one texture is sufficient.

c. When the vertices of a single triangle have different colors, the interior color of the triangle will linearly interpolate between those colors, unless a triangle color has been explicitly specified (because a triangle color takes precedence over a vertex color). When two triangles specify different colors to the same vertex, the colors will be averaged.

K. Transparency

a. The transparency channel <a> determines alpha compositing for combining the specified foreground color with a background color to create the appearance of partial transparency. A value of zero specifies zero transparency, that is, only the foreground color is used. A value of one specifies full transparency, that is, only the background color is used. Intermediate values are used for a linear combination.

b. Negative values are rounded to zero and values greater than one are truncated to one. The background color of a triangle is the vertex color. The background color of a vertex is the volume color, then object, and material in decreasing precedence.

L. Texture Specification

a. The <texture> element can be used to associate a texture id with a particular texture data. The texture map size will be specified and both two-dimensional (2D) and 3D maps are supported. The data will be an encoded string of bytes in Base64 encoding as grayscale values. Grayscale will be encoded as a string of individual bytes, one per pixel, specifying the grayscale level in the 0-255 range. The ordering of data will start with the top left corner and proceeding left to right then top to bottom. A 3D texture will specify the first layer initially and repeat for all subsequent depths. The data will be truncated or appended with zero values as needed to meet the specified texture size.

M. Print Constellations

a. Multiple objects can be arranged together using the <constellation> element. A constellation can specify the position and orientation of objects to increase packing efficiency and describe large arrays of identical objects. The <instance> element specifies the displacement and rotation an existing object needs to undergo into its position in the constellation. The displacement and rotation are always defined relatively to the original position and orientation in which the object was originally defined. Rotation angles are specified in degrees and are applied first to rotation about the *x* axis, then the *y* axis, and then the *z* axis.

b. A constellation can refer to another constellation. However, recursive or cyclic definitions of constellations are not allowed. When multiple objects and constellations

are defined in a single file, only the top level objects and constellations are available for printing.

c. The orientation at which the objects will be printed will default to those specified in the constellation. The *z* axis is assumed to be the vertical axis, with the positive direction pointing upwards and zero referring to the printing surface. The *x* and *y* directions will correspond to the main build stage axes if a gantry positioning system is used.

N. Metadata

a. The <metadata> element can optionally be used to specify additional information about the objects, geometries, and materials being defined (see slide 9). For example, this information can specify a name, textual description, authorship, copyright information, and special instructions. The <metadata> element can be included at the top level to specify attributes of the entire file or within objects, volumes, and materials to specify attributes local to that entity.

O. Compression and Distribution (see slide -10)

a. An AMF shall be stored either in plain text or be compressed. If compressed, the compression shall be in ZIP archive format and can be done manually or at write time using any one of several open compression libraries.

b. Both the compressed and uncompressed version of this file will have the AMF extension, and it is the responsibility of the parsing program to determine whether or not the file is compressed and if so, to perform decompression during read.

c. Additional files may be included in the ZIP archive such as manifest files and electronic signatures. However, only the AMF file with the same name as the archive file will be parsed. Absence of a file with that name will constitute an error.

d. This specification does not specify any explicit mechanisms for ensuring data integrity, electronic signatures, and encryptions.

P. Tolerances, Textures, and Additional Information

a. It is recognized that there is additional information relevant to the final part that is not covered by the current version of this specification.

b. Prototyping
c. Modeling
d. Production parts

C. Post Processing
   a. Infusion

D. File Size Comparison (see slide 12)

Evaluation:

1. Describe the .amf file.
2. What advantages does an .amf file have over a .stl file?
3. What additional information can be included in an .amf file?

References:

1. The core competencies used in the development of this module were taken from the draft of the Additive Manufacturing Core Competencies for Technicians soon to be released by The National Resource Center for Materials Technology Education (MatEdU), Technician Education in Additive Manufacturing (Project TEAM), under NSF Grant: DUE #1003530

2. Standard Specifications for Additive Manufacturing File Format (.AMF), ASTM Standard ISO/ASTM 52915-13