Appendix V
Deliverable #5
SCAN Schema Documentation

88 ABW Case Completed: Case Number 88ABW-2017-3854

Project 4039

Development and Demonstration of Open Source Protocols for Powder Bed Fusion Additive Manufacturing (PBFAM)

Report Released: September, 2016

Submitted by: William T. Carter
Additive Manufacturing Laboratory
GE Global Research

NCDMM
NATIONAL CENTER FOR DEFENSE
MANUFACTURING AND MACHINING

America Makes
National Additive Manufacturing Innovation Institute

486 Cornell Road
Suite 2
Blairsville, PA 15717
Phone: (724) 539-8811
www.ncdmm.org

"The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Government."

Distribution authorized to U.S. Government Agencies and America Makes Members; Critical Technology; (START DATE OF EFFORT). Other request for this document shall be referred to AFRL/RXMS, Wright-Patterson Air Force Base, OH 45433-7750.

Authorship

GE Global Research	*GE Aviation*	*Lawrence Livermore National Laboratory*
- William Carter - Charles Gilman - Kirk Mathews - Subhrajit Roychowdhury	- Ken Hix - Justin Mamrak	- Manyalibo Matthews - Gabe Guss

Introduction

During 2015 and 2016, GE Global Research led an America Makes program including GE Aviation's Additive Development Center (ADC) and the Lawrence Livermore National Laboratory (LLNL) to establish a LAYER Protocol and a SCAN Protocol, and to demonstrate the efficacy of each by generating parts from STL files on PBFAM machines.

This document gives the details of the SCAN protocol, which is implemented in the XML (eXtensible Markup Language) format.

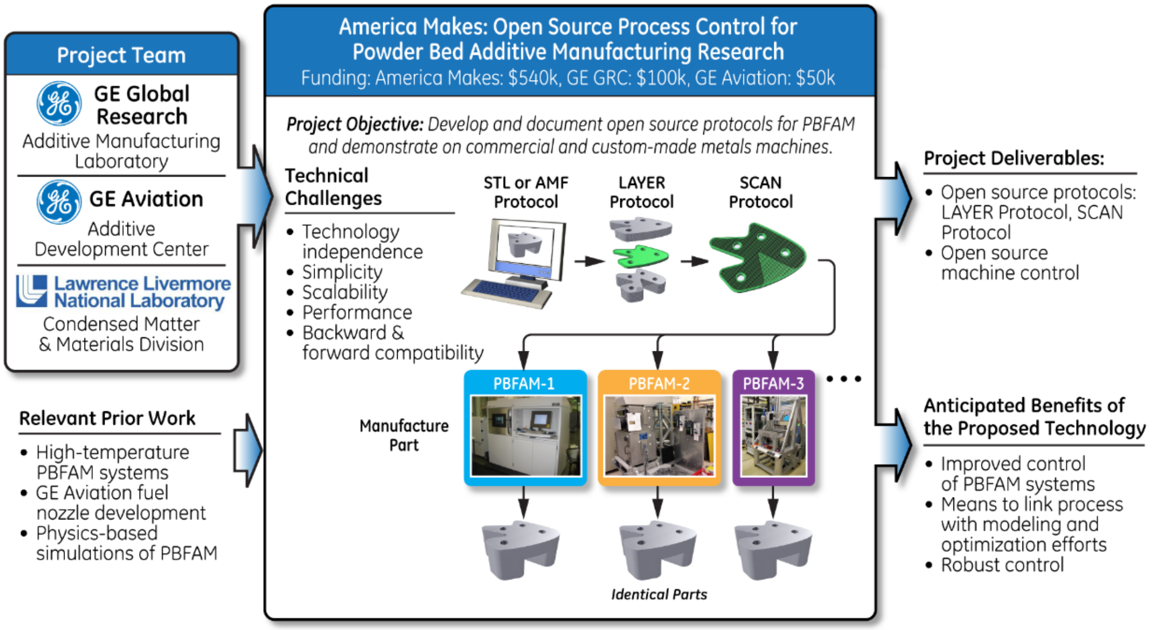
Project Team
GE Global Research
Additive Manufacturing Laboratory
GE Aviation
Additive Development Center
Lawrence Livermore National Laboratory
Condensed Matter & Materials Division
Relevant Prior Work
• High-temperature PBFAM systems
• GE Aviation fuel nozzle development
• Physics-based simulations of PBFAM
America Makes: Open Source Process Control for Powder Bed Additive Manufacturing Research
Funding: America Makes: $540k, GE GRC: $100k, GE Aviation: $50k
Project Objective: Develop and document open source protocols for PBFAM and demonstrate on commercial and custom-made metals machines.
Technical Challenges
• Technology independence
• Simplicity
• Scalability
• Performance
• Backward & forward compatibility
STL or AMF Protocol
LAYER Protocol
SCAN Protocol
Manufacture Part
PBFAM-1
PBFAM-2
PBFAM-3
Identical Parts
Project Deliverables:
• Open source protocols: LAYER Protocol, SCAN Protocol
• Open source machine control
Anticipated Benefits of the Proposed Technology
• Improved control of PBFAM systems
• Means to link process with modeling and optimization efforts
• Robust control

Figure 1 Project Overview

Scope

This specification defines a framework for an interchange format to address the current and future needs of powder bed additive manufacturing (PBFAM). There exist several standards that define the geometry and properties of the final object or part to be built by the PBFAM system, e.g. STL, AMF. Traditionally, these geometry files are processed by software developed by the PBFAM manufacturer. This processing software is closed – the internal formats are not available to the user and the data cannot be adjusted by the user. If the intended part is not produced, the user does not have the means to debug the manufacturer's processing software and often resorts to modifying the initial geometry file to achieve the desired results. This LAYER file specification allows the processing software to decouple geometry processing from machine parameters. The process sequence of the PBFAM project developed is illustrated in Figure 2. The geometry file is processed or sliced into layers. The layers are stored in one or more LAYER files. The layer files contain the layer geometry as well as other properties that may be useful in generating the laser scan paths to create the part. The LAYER files are processed by the Scan Generator and a series of SCAN files are created. The SCAN files contain geometry related to the laser position and scan parameters for each layer. The SCAN files are used to direct the Machine Controller.

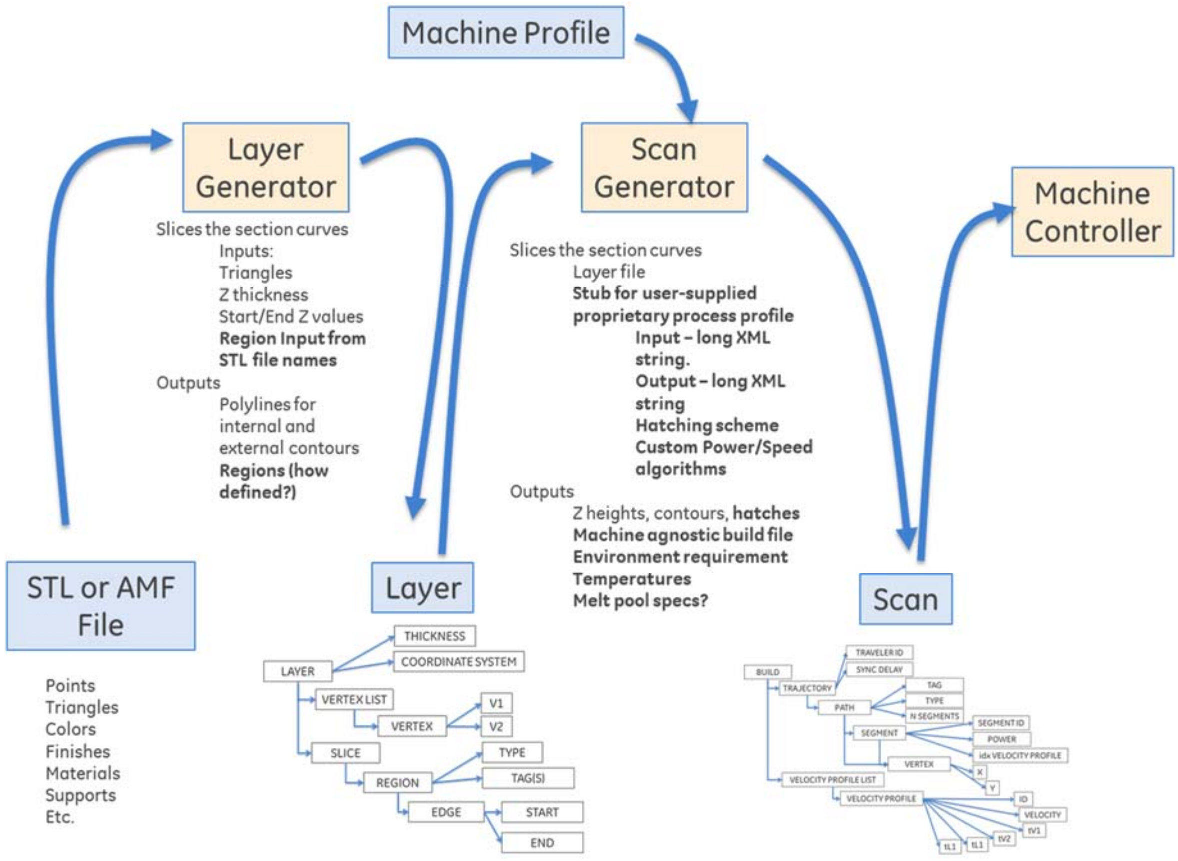

Figure 2 The PBFAM Architecture

The LAYER files are stored in the XML (eXtensible Markup Language) format. They are intended to be both human and machine readable and the visualizers and editors for this format may be readily developed. The content of the SCAN file is illustrated in Figure 3 and given in more detail in Table 1.

The LAYER files will be read by the Scan Generator software which will process the layers and create SCAN files. These SCAN files will be stored in XML (eXtensible Markup Language) format. They are intended to be both human and machine readable and the visualizers and editors for this format may be readily developed.

This document describes the XML types and elements. The schema is given in the Appendix.

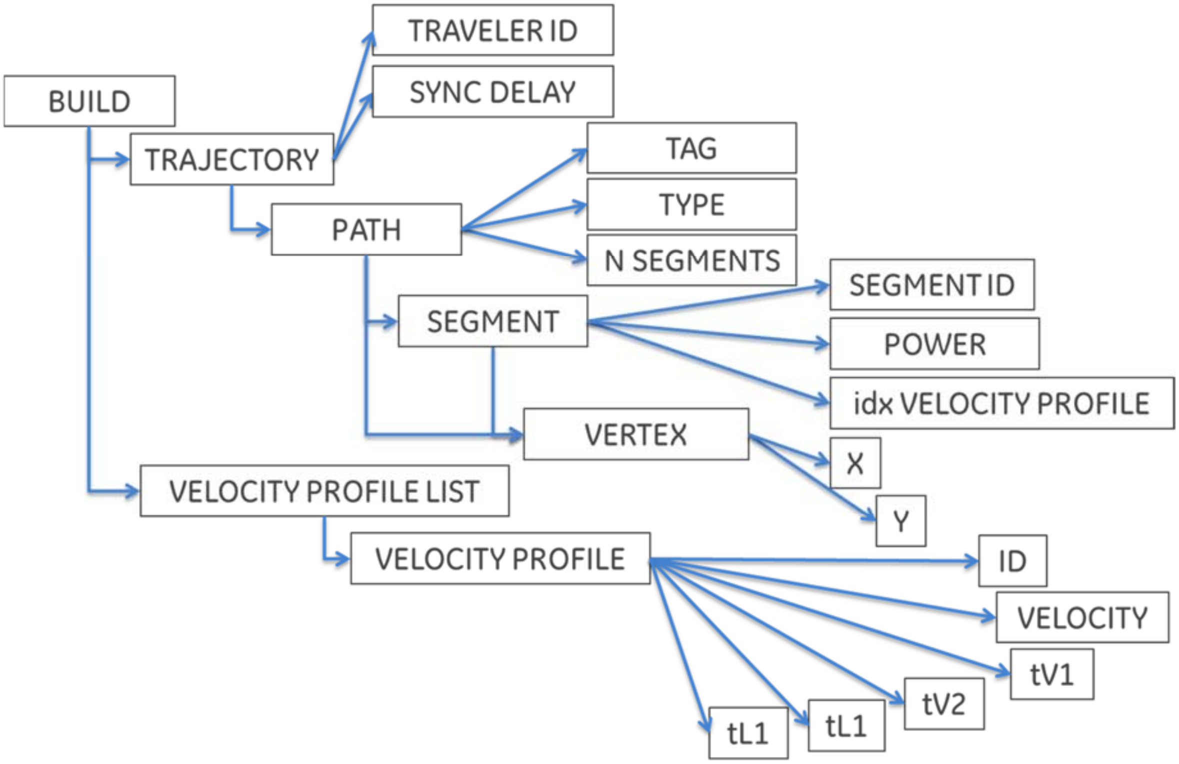

Figure 3 Contents of the SCAN File

Table 1: Contents of the SCAN Schema

Element	Parent Element(s)	Elements
BuildType	None	VelocityProfileList, Trajectory,
VelocityProfileListType	BuildType	VelocityProfile
VelocityProfileType	VelocityProfileList	ID, Velocity, tV1, tV2, tL1, tL1
TrajectoryType	BuildType	TravelerID, SyncDelay, Path
PathType	TrajectoryType	Type, Tag, NumSegments, Segment,
Vertex	PathType, SegmentType	X, Y
SegmentType	PathType	SegmentID, Power, idxVelocityProfile, End

SCAN Schema Elements

In the XML schema, types have elements and attributes. The elements used in the SCAN XML file are described in this section. The elements used in the SCAN file schema are:

- BuildType,
- VelocityProfileListType,
- VelocityProfileType,
- TrajectoryType,
- PathType,
- Vertex, and
- SegmentType.

BuildType

The BuildType is the container for the machine parameters and scan paths necessary to produce one layer of a three-dimensional part. The BuildType has two elements VelocityProfileList – a container for all of the Velocity Profiles used in the build (layer) – and an unbounded list of the element Trajectory – which describes a single point to point laser path.

VelocityProfileListType

The VelocityProfileListType is an ordered list that contains all of the VelocityProfiles used in the scan. It has a single element VelocityProfileList that can occur one or more times.

VelocityProfileType

The VelocityProfileType holds the information for a single velocity profile used by the laser in the scan. It has several elements: ID – a string that holds a unique value for the velocity profile; Velocity – the mark or jump velocity in millimeters per second; Mode – an enumerated value that is either delay or acceleration; and tV1, tV2, tL1, tL1 – rise and fall times in acceleration mode (tV1, tV2) and laser on and off delay times (tL1 and tL1), all times in micro seconds.

TrajectoryType

The TrajectoryType is the scan path for the laser. There is one TrajectoryTpe per laser, i.e. multiple laser systems will support multiple trajectories. The TrajectoryType has the following elements: TravelerID – a unique identifier for the laser associated with the trajectory; SyncDelay – parameter to support synchronization of a multi-laser system in micro seconds; Path – one or more routes that the laser uses to complete the trajectory.

PathType

The PathType is a collection of curve or line segments. Each segment in the path corresponds to a mark or jump move. The PathType is composed of the following elements: Type – a description of the type of path, two examples of this are hatch or contour; Tag – a user defined description inherited from the layer file; NumSegments – an integer describing the number of segments in the path; Start – the point that represents the starting point for the path; and Segment – one or more component curves or line segments and the laser parameters that make up the bulk of the path.

Vertex

The VertexType is a point in a standard x-y Cartesian coordinate system. The VertexType contains two elements: X – the X location in the coordinate system; and Y – the Y location in the coordinate system.

SegmentType

The SegmentType contains the laser parameters and location for a mark or jump move for the laser. The SegmentType contains the following elements: SegmentID – has a default value of 0 and can be used with special features; Power – the laser power in Watts; idxVelocityProfile – the index value in the VelocityProfileList of the laser velocity profile; and End – the end point of the Segment.

Appendix
SCAN Schema Version 2 XSD

```
<!--                                                              -->
<!-- PBFAM SCAN Protocol XML Schema v2.0                          -->
<!-- Author(s): W. Carter, S. Roychowdhury                        -->
<!-- and PBFAM team                                               -->
<!-- Date: 05 May 2016                                            -->
<!-- Copyright 2016 The General Electric Company                  -->
<!--                                                              -->
<!-- need URL to host this XML schema for GE/America Makes        -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:annotation>
        <xsd:documentation xml:lang="en">
            PBFAM SCAN Protocol
            Copyright 2016
            ***Legal Language Needs to be inserted here***
            </xsd:documentation>
    </xsd:annotation>
    <!--                                                   -->
    <!-- Element Definitions                               -->
    <!-- All of these elements can be top level elements   -->
    <!-- in the XML file. This might change in the next    -->
    <!-- version.  This is for ease of definition          -->
    <!--                                                   -->

    <xsd:element name="Build" type="BuildType"/>
    <xsd:element name="VelocityProfileList" type="VelocityProfileListType"/>
    <xsd:element name="VelocityProfile" type="VelocityProfileType"/>
    <xsd:element name="Trajectory" type="TrajectoryType"/>
    <xsd:element name="Vertex" type="VertexType"/>
    <xsd:element name="Path" type="PathType"/>
    <xsd:element name="Segment" type="SegmentType"/>

    <!-- Type Definitions                               -->

    <xsd:complexType name="VertexType">
        <xsd:sequence>
            <xsd:element name="X" type="xsd:decimal"/>
            <xsd:element name="Y" type="xsd:decimal"/>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="SegmentType">
        <xsd:sequence>
            <xsd:element name="SegmentID" type="xsd:string" />
            <xsd:element name="Power" type="xsd:decimal"/>
            <xsd:element name="idxVelocityProfile" type="xsd:integer"/>
            <xsd:element name="End" type="PointType" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="PathType">
        <xsd:sequence>
```

```
            <xsd:element name="Type" type="xsd:string" />
            <xsd:element name="Tag" type="xsd:string" />
            <xsd:element name="NumSegmants" type="xsd:integer" />
            <xsd:element name="Start" type="PointType" />
             <xsd:element   name="Segment"   type="SegmentType"   minOccurs="1"
maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="VelocityProfileType">
        <xsd:sequence>
            <xsd:element name="ID" type="xsd:string" />
            <xsd:element name="Velocity" type="xsd:decimal" />
            <xsd:element name="Mode" type="xsd:string" />
            <xsd:element name="tV1" type="xsd:decimal" />
            <xsd:element name="tV2" type="xsd:decimal" />
            <xsd:element    name="tL1"    type="xsd:decimal"    minOccurs="2"
maxOccurs="2" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="VelocityProfileListType">
        <xsd:sequence>
            <xsd:element   name="VelocityProfile"   type="VelocityProfileType"
minOccurs="1" maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="TrajectoryType">
        <xsd:sequence>
            <xsd:element name="TravelerID" type="xsd:string" />
            <xsd:element name="SyncDelay" type="xsd:decimal" />
            <xsd:element     name="Path"     type="PathType"     minOccurs="1"
maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="BuildType">
        <xsd:sequence>
            <xsd:element                            name="VelocityProfileList"
type="VelocityProfileListType" />
            <xsd:element name="Trajectory" type="TrajectoryType" minOccurs="1"
maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>
</xsd:schema>
```