OGC Testbed-14

Point Cloud Data Handling Engineering Report

Table of Contents

Publication Date: 2019-03-08

Approval Date: 2019-02-28

Submission Date: 2018-11-01

Reference number of this document: OGC 18-048r1

Reference URL for this document: http://www.opengis.net/doc/PER/t14-D013

Category: Public Engineering Report

Editor: Howard Butler

Title: OGC Testbed-14: Point Cloud Data Handling Engineering Report

---

OGC Engineering Report

COPYRIGHT

Copyright (c) 2019 Open Geospatial Consortium. To obtain additional rights of use, visit http://www.opengeospatial.org/

WARNING

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements. However, the discussions in this document could very well lead to the definition of an OGC Standard.

LICENSE AGREEMENT

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD. THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to

indicate compliance with any LICENSOR standards or specifications.

This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

Chapter 1. Summary

This Engineering Report (ER) describes requirements that a point cloud web service must satisfy to enable application developers to provide convenient remote access to point clouds. It provides a short contrast of five point cloud web service software approaches (Esri I3S [https://github.com/esri/pointcloud], 3D Tiles [https://github.com/AnalyticalGraphicsInc/3d-tiles], Greyhound [https://greyhound.io], PotreeConverter [https://github.com/potree/PotreeConverter], and Entwine [https://entwine.io]) and their implementations available at the time of the report. A small industry survey about these requirements is also provided in support of the report's discussion about formats, web service requirements, industry support, and industry desire on these topics.

1.1. Business Value of ER

By providing a workable set of point cloud web service requirements, this report gives follow-on efforts direction in regard to the challenges that future standardization efforts must tackle. The report also links together a survey snapshot of the current industry software situation in regard to both prevalent formats and web services.

1.2. Key findings

- Description of a point cloud web service
- Listing of required and optional features of a point cloud web service
- Market status for point cloud web service software

1.3. Requirements & Research Motivation

Before standardization of any web services approaches can be undertaken, exploration of the software requirements that such a service must meet would provide helpful background to achieve the task for the widest implementation audience possible. This report seeks to accomplish this task in five ways. First, a description of point cloud web service requirements is presented in Chapter 6 along with how they relate to current industry software usage. Second, existing point cloud web service approaches are contrasted in Chapter 7 to the requirements list and discussion about their consequences is provided. Third, discussion about the approaches, the consequences of some of their design choices, and their relationship to the requirements presented in Chapter 6 is provided in Chapter 8. Lastly, the results of a short industry survey in Industry Survey provide background on the niche for point cloud web services, point cloud formats, and the desire for compression on both topics.

1.4. Research Motivation

Point cloud data management techniques could greatly benefit from a web services approach, due to their imposing bulk and fundamental difference versus typical raster and vector geospatial data. A point cloud web service provides an opportunity for application developers to offload the challenges associated with data compression, partitioning, and information extraction. A web service approach also provides opportunity for applications to leverage the same data access and organization approach, which allows applications to focus on providing upstream value such as feature extraction, data fusion, and monitoring.

Questions that were addressed by research efforts into point cloud web services include the following:

- What is unique about point cloud data in relation to other geospatial data?
- Why is a web service for point clouds needed?
- What is the expected data volume of point cloud data?
- Do existing OGC web service standards already address the point cloud challenge?
- Describe the challenges of point cloud web services in relation to other kinds of web services such as those based on Web Map Service (WMS), Web Feature Service (WFS), and Web Coverage Service (WCS) standards

A service-based approach to point cloud data access and dissemination is a missing capability required for interoperability within the point cloud data industry. A service-based approach has the potential to allow for separation of concerns, by especially making distinct the problem of point cloud data organization from access challenges.

This report describes and discusses requirements for point cloud web services, it contrasts them in relation to five point cloud web service implementations, it highlights the technical details of an example specification, and it provides the results of an industry survey on the topic of point cloud web services and formats.

1.5. Recommendations for Future Work

Specific recommendations that could follow the information provided in this report include:

- The Point Cloud DWG should solicit additional examples and candidate clients of point cloud web services and compare those to the Required and Optional requirements defined in this report.
- OGC should begin the Standards Working Group (SWG) process to coalesce a group around the construction of a static point cloud web service modeled on some of the same principles of OGC Web Map Tile Service (WMTS) and the candidate OGC Tile Matrix Set Standards. The SWG should identify common principles for an OGC point cloud web service standard. These questions include answering the following questions:
    a. Should a point cloud web service utilize replacement or additive refinement?
    b. Should a point cloud web service have implicit or explicit structure?
    c. Should a point cloud web service mandate a content encoding and compression approach?
- The survey in Appendix A illustrates the market is still missing an openly specified format with the following features:
    a. Spatially accelerated access
    b. Point cloud-appropriate compression with no intellectual property restrictions
    c. Flexible data schema

d. Platform (JavaScript, Java, native) flexibility

1.5.1. Recommended Future Tasks

- Conduct an engineering analysis of point cloud web services to evaluate performance and behavior trade-offs for explicitly designed workflows. These might include specific over-the-network data rendering scenarios and achieve retrieval in support of data processing.
- Develop a performance proofing environment for point cloud web services.
- Construct a point cloud web service verification testbed.
- Collaborate with the American Society for Photogrammetry and Remote Sensing (ASPRS) to find a future point cloud format solution that is well aligned with efforts such as WKTv2, WMTS, and GeoPackage.

1.6. Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Contacts

Name	Organization
Howard Butler	Hobu, Inc.
Connor Manning	Hobu, Inc.
Jean–François Bourgon	Natural Resources Canada
Charles Heazel	WiSC

1.7. Foreword

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

Chapter 2. References

The following normative documents are referenced in this document.

- OGC: OGC 06-121r9, OGC® Web Services Common Standard, 2010 [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2]
- OGC: OGC 09-110r4, OGC® Web Coverage Service Standard, 2012 [https://portal.opengeospatial.org/files/09-110r4]
- OGC: OGC 07-057r7, OGC® Web Map Tile Service Implementation Standard, 2010 [https://portal.opengeospatial.org/files/?artifact_id=35326]
- OGC: OGC 12-063r5, OGC® Geographic information - Well-known text representation of coordinate reference systems, 2015 [http://docs.opengeospatial.org/is/12-063r5/12-063r5.html]
- OGC: OGC 12-128r14, OGC® Geopackage Encoding Standard, 2017 [https://portal.opengeospatial.org/files/12-128r14]
- OGC: OGC 17-030r1, LAS Specification 1.4 OGC® Community Standard, 2018 [https://portal.opengeospatial.org/files/17-030r1]
- OGC: OGC Indexed 3d Scene Layer (I3S) and Scene Layer Package Format Specification, 2017 [http://docs.opengeospatial.org/cs/17-014r5/17-014r5.html]
- NGA: NGA.SIG.0020_1.0_BPF. Binary Point File 3 (BPF3), BPF Public License File Format Definition, Implementation Guide, 2015 [https://nsgreg.nga.mil/doc/view?i=4202]

Chapter 3. Terms and definitions

IMPORTANT

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard OGC 06-121r9 [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2] shall apply. In addition, the following terms and definitions apply.

3.1. Terms

- **pointcloud**

  A set of points in three dimensional space with an optional set of supporting attributes such as intensity, color information, or time.

- **streaming**

  In the context of point clouds, the ability of software systems to make partial requests for data segmented by geography or attributes over a network.

- **LiDAR**

  Light Detection and Ranging — a common method for acquiring point clouds through aerial, terrestrial, and mobile acquisition methods.

- **octree**

  A recursive partitioning of three-dimensional space into octants [1], [2].

3.2. Abbreviated terms

- RTT Round trip time. The total amount of time a request for a data resource takes.

Chapter 4. Overview

Chapter 5 describes properties of point cloud data that make it somewhat unique in relation to typical vector or raster geospatial data.

Chapter 6 defines and describes the requirements a point cloud web service should satisify.

Chapter 7 contrasts five point cloud web service implementations available for evaluation at the time of writing.

Chapter 8 provides discussion and summary information about point cloud web services, identifies some fruitful topics to pursue in the domain, and discusses the current state-of-the-art on the topic.

Appendix A summarizes a short industry survey undertaken by the authors in the Fall of 2018 on the topics of point cloud web services, formats, and point cloud compression.

Chapter 5. Introduction

A point cloud is a data type representing discrete locations in a three-dimensional space, with optional storage of subsequent attributes such as reflectance, capture time, color information, or multi-return status. In geospatial contexts, point clouds are often captured through actively-sensed systems such as Light Detection and Ranging (LiDAR) scanners or computed from coincidence-matched imagery to support the characterization of volumes.

Point cloud data has become more prevalent due to accelerating computing hardware capability. Lower cost computation has meant that measurement activities through laser scanners and image processing that was once too computationally intensive is now viable from a cost and practicality perspective. Scanners and photography continue to increase the density and volume of data being captured, and the need for data management and access tools that can handle massive collections of data continues to increase.

5.1. Data Properties

The primary challenge of point cloud data is simply their storage volume. Capture rates for geospatial point cloud data, in the form of actively-captured LiDAR or passively-captured coincidence-matched imagery are still accelerating in 2018 as sensors, storage, and processors continue their rapid increase in capability and decrease in cost.

The data type does not lend itself to convenient compression, storage arrangement, or access mechanisms – especially in relation to other typical geospatial data types. The data behave as not quite vector and not quite imagery, and point cloud data need to employ data management techniques that address the unique challenges they present – specifically the requirement of per-point storage of coordinates and attributes.

5.1.1. Vector Data Similarities

Point cloud data have properties in common with typical geospatial vector data. They are often used as a frame for point to point measurement, and each point might have additional attributes associated with its capture such as a capture time, reflectance, return information, color, or incidence angle. A typical geospatial vector database would make these supporting attributes conveniently available for data segmentation activities, but their volume presents challenges. For example, it would not be unreasonable for a very high detail aerial scene capture of ten square kilometers to have a capture volume of ten billion points. A table with ten billion points would stretch the capabilities of most geospatial database software, and the overhead associated with indexing and storage of that data would quickly become prohibitive.

Point cloud data are frequently stored in similar organizations as vector data. The OGC Community Standard ASPRS LAS (OGC 17-030r1) format, for example, models each point individually, stored in an order that may or may not have meaning, with each X, Y, and Z coordinate completely specified. Other formats such as Binary Point Format (BPF) (NGA.SIG.0020_1.0_BPF) of the National Geospatial Intelligence Agency (NGA) also completely model coordinates and attributes. Both formats do not natively support accelerated access via spatial partitioning, but BPF does have a compression story that is sufficient for many scenarios whereas compression for LAS is found in the add-on open source library and encoding called LASzip [http://laszip.org].

5.1.2. Raster Data Similarities

Similar to some raster data, point cloud data are often an equidense surface of measurements slightly perturbed in location. Much like an aerial or satellite image they most often represent a time-integrated "capture" of a scene. Point cloud data are almost never transacted like typical geospatial data in the sense that their positions are not changed or edited after final quality control and processing of the capture. Additional application of summary attributes, such as assigning of new classification data or ancillary downstream derivatives such as incidence normals is commonly applied to data.

Voxelized storage, where points are associated with cells of a divided 3D space, are frequently used to represent storage of point clouds as a type rasterized geospatial data. A disadvantage of voxelization is the need to summarize data to a specific resolution, which can have the effect of failing to capture the full fidelity of the data. This can make voxelized storage approaches unsuitable for archival storage, even if it makes a convenient organization for many exploitation or processing activities.

5.2. Data Uses

Point cloud data are often treated as raw measurements in support of upstream data extraction and production scenarios. These subsequent usages include other domains such as:

- Modeling and simulation
- Measurement
- Feature extraction
- Change detection
- Bathymetric exploration

In nearly all scenarios, point cloud data are processed into a derived product for exploitation or information extraction. For example, it is common to rasterize point cloud data to a two-and-half-dimensional (2.5D) surface model to allow the data to be exploited by typical landscape morphology operations. Alternatively, feature extraction algorithms might segment and construct solid surfaces by connecting points together. In the first scenario, simple range query support is sufficient to allow an application to quickly produce the derived output. In the second, hierarchical spatial indexing with accelerated per-point access is needed to support the neighborhood searching required for the segmentation and classification process.

5.3. Data Transmission

A typical scenario for data processing is to take a full density subset of a larger point cloud collection, process or extract information from it, and continue to the next processing step. Because the data are typically so large, transmission of subsets over networks is a critical requirement that most applications require – even in cloud computing scenarios. Data organization properties that can improve transmission of point cloud data include efficient compression, idempotent responses that allow repeated requests to return the same data, and service architectures that do not require significant computing to respond to each request.

Network protocols for point data transmission must account for hierarchical data access with a query pattern that supports requesting level-of-detail subsets of data. Without this ability, full-density computation operations have a tendency to consume extra computation due to the fact that individual points in a larger point cloud do not contribute significant informational weight. By selecting for data at specific resolutions, applications can balance their need for precision with network and processing performance.

While many organizational approaches are possible, two indexing patterns are prevalent in point cloud data operations – a *k-style* index, which supports rapid access neighborhood lookups and quadtree or octree indexes, which support range queries by depth. While each have their use, octree indexing patterns provide the widest applicability and utility for services that seek to do partial extraction of spatial windows of data. Query patterns that require neighborhood searches can be achieved by a multi-pass or sub-window secondary indexing.

NOTE

An important consideration of data transmission challenges for point cloud data is balancing data fidelity with data size. A typical usage scenario for LiDAR data is precise geopositioning, and data manipulation such as position quantization made in the name of transmission efficiency can cascade aliasing effects downstream through a point cloud processing workflow. These consequences must be considered before altering data in service of expedient transmission approaches.

5.4. Data Formats

Numerous data formats are prevalent in the point cloud data domain, and they tend to be segmented between "geospatial" point cloud formats and those that are unconcerned with world positioning. As confirmed by the question in Figure 9 of Appendix A, the OGC Community Standard ASPRS LAS format is likely the most commonly used format in the geospatial community. Its compressed, open source content encoding is called LASzip [https://laszip.org] (or LAZ). Combined, these two formats dominate the marketplace for interchange and archive of geospatial point clouds.

5.4.1. File Formats

Non-geospatial point cloud file formats, especially ASCII and PLY, are common in situations closer to sensors or closer to visualization. Open source software exists for LAS, LAZ, PLY, BPF, and many of the formats described in Table 1. The formats listed span a wide array of primary features. Some focus on compression and archive, while others support accelerated spatial access. Many require a fixed data schema, but some support a flexible definition. As of 2018, LAS and LAZ seem to have provided the widest interoperability story for geospatial point cloud formats, but questions about LAS in the Industry Survey show that the market is still not completely satisfied with the feature set of the most commonly used data storage encoding.

Table 1. Geospatial Point Cloud Formats

Extension	Common Name	Open Source Implementation?
3DM	Rhinoceros file format [https://en.wikipedia.org/wiki/Rhinoceros_3D]	Yes
ASCII	Plain text	Yes
BIN	TerraSolid .BIN [https://www.terrasolid.com/download/tscan.pdf]	Yes
BPF	Binary Point Format [https://nsgreg.nga.mil/doc/view?i=4202]	Yes
CSD	Optech Corrected Sensor Data [https://pdal.io/stages/readers.optech.html]	Yes
CL3	Topcon CL3 and CLR [https://www.topconpositioning.com/sites/default/files/product_files/topcon_.cl3_point_cloud_file_format.pdf]	No
E57	ASTM E2807 [http://www.libe57.org/]	Yes
EPT	Entwine [https://entwine.io]	Yes
Faro	Faro FWS and FLS [https://knowledge.faro.com/Software/FARO_SCENE/SCENE/SDK_File_Download_for_SCENE]	No
FBI	TerraSolid FASTBINARY .FBI [https://www.terrasolid.com/download/tscan.pdf]	No
HDF	Hierarchical Data Format [https://support.hdfgroup.org/HDF5/whatishdf5.html]	Yes
I3S	Esri Indexed 3d Scene Layer [https://github.com/esri/pointcloud]	Yes
ILVIS2	NASA IceBridge ILVIS2 [https://nsidc.org/data/ilvis2]	Yes
LAS	ASPRS LAS File Format [https://www.asprs.org/divisions-committees/lidar-division/laser-las-file-format-exchange-activities]	Yes
LAZ	Compressed LAS [https://laszip.org]	Yes
MIPC	Modality Independent Point Cloud [https://slideplayer.com/slide/11259045/]	No
MrSID	Extensis (LizardTech) MrSID LiDAR [http://doc.extensis.com/docs/SDK/MrSID%20Decode%20SDK%20for%20LiDAR%20User%20Manual.pdf]	No
NITF	National Imagery Transmission Format [https://pdal.io/stages/readers.nitf.html]	Yes
OPALS	OPALS Generic Format [https://geo.tuwien.ac.at/opals/html/ref_fmt_gen.html]	Yes
PCD	Point Cloud Data [http://pointclouds.org/documentation/tutorials/pcd_file_format.php] (PCL [http://pointcloud.org])	Yes
PLY	Standford Triangle Format [http://paulbourke.net/dataformats/ply/]	Yes

Extension	Common Name	Open Source Implementation?
POD	Bentley POD [https://www.bentley.com/en/products/product-line/reality-modeling-software/bentley-pointools]	No
PTS	Cyclone PTS [https://w3.leica-geosystems.com/kb/?guid=5532D590-114C-43CD-A55F-FE79E5937CB2]	Yes
PTX	Cyclone PTX [https://w3.leica-geosystems.com/kb/?guid=5532D590-114C-43CD-A55F-FE79E5937CB2]	Yes
RXP	RIEGL RXP [http://www.riegl.com/]	No
RDB	RIEGL RDP [http://www.riegl.com/]	No
SLPK	Esri Indexed 3d Scene Layer (packaged) [https://github.com/esri/pointcloud]	Yes
zLAS	ESRI zLAS [https://www.esri.com/arcgis-blog/products/product/3d-gis/esri-optimized-las-zlas-i-o-runtime-library-is-now-available/]	No

5.4.2. Database Formats

Point cloud data storage solutions exist for three common relational database systems. Oracle pioneered the topic with their Oracle Point Cloud storage solution, and a similar data schema and metadata structure was followed by PostgreSQL and SQLite. Read and write software support for all three databases is available in the open source PDAL [https://pdal.io] software package.

Table 2. Database-oriented Point Cloud Formats

Database	Open Source Implementation?
Oracle Point Cloud [https://docs.oracle.com/cd/B28359_01/appdev.111/b28400/sdo_pc_pkg_ref.htm]	No
pgpointcloud [https://github.com/pgpointcloud/pointcloud/]	Yes
SQLite [https://pdal.io/stages/writers.sqlite.html]	Yes

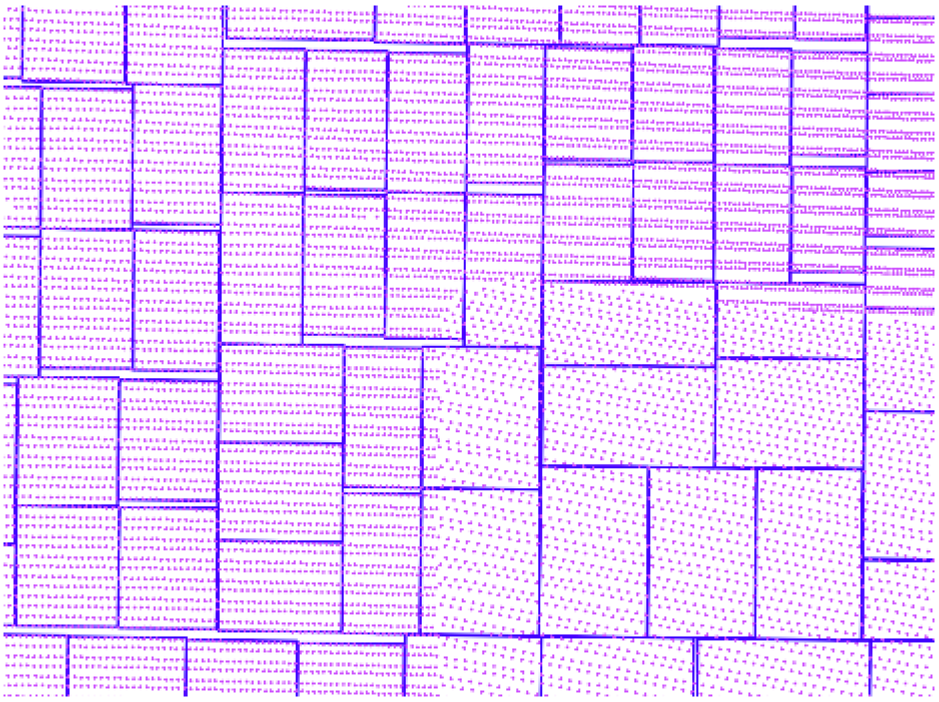


Figure 1. PDAL [https://pdal.io] adapted and permuted a data organization approach first constructed by the Oracle team called chipper [https://pdal.io/stages/filters.chipper.html] that seeks to balance data storage to a specified page size without excessive exact recursion. The storage approach reduces full-resolution data over-selection, but it does not address the concept of selection of data to a specified resolution. PDAL allows chipper-organized data to be stored in all three relational database systems described in Table 2.

Chapter 6. Web Services Requirements

Given the desired use, storage, and transmission of point cloud data, software systems that abstract access to data via network access could provide for convenient separation of concerns. Applications could control the content and organization of data they request and services could focus on data delivery, storage, and organization.

This section provides a list of requirements for point cloud web services and discusses some consequences of them being provided by point cloud web service software. Some of the requirements listed in this section are categorized as "Required" in the sense that a point cloud web service without an answer to it might not be credible. Requirements in the "Optional" section are certainly nice-to-have, but in most situations a point cloud web service could function without them.

6.1. Required

Software should meet the following list of requirements to provide a credible capability stack for a point cloud web service. "Required" features share the property that downstream client applications expect to be able to depend upon them to support their application’s usage of point cloud data through a point cloud web service. While some of the requirements may be more important than others, no distinction is made in this report beyond describing their need.

6.1.1. Flexible Schema

A point cloud web service should be agnostic to data format, and it must support the flexible definition of content, much like a typical geospatial vector database. They must be able to consume data in a variety of point cloud formats, including ASPRS LAS 1.0 to 1.4, NGA BPF, PLY, ASCII, and database approaches such as pgpointcloud [https://github.com/pgpointcloud/pointcloud] or Oracle Point Cloud [https://docs.oracle.com/cd/B28359_01/appdev.111/b28400/sdo_pc_pkg_ref.htm].

Points should be allowed to be stored as fixed precision integers, floating point single precision numbers, and floating point double precision numbers. Any valid primitive data type (integers, strings, etc). should be allowed for attribute information. The requirement does not mandate how the service is to approach meeting the requirement so long as the external web service supports these options.

6.1.2. Configurable Data Sources

Some situations would model point cloud data as a single contiguous data resource. One example might be a realized global surface of the data from the new NASA ICESat2 [https://icesat-2.gsfc.nasa.gov/] satellite. Other data organizations might have different demands. LiDAR data captured by the USGS 3DEP [https://www.usgs.gov/core-science-systems/ngp/3dep] program contains a patchwork quilt of data captures by administrative unit and funding source. A credible point cloud web service solution must be able to combine, process, and extract data from this spectrum of configurations. Clients must be able to merge, combine, and process data from multiple service endpoints, and clients must be able to control the resources and their data mixture, should there be any.

6.1.3. Spatial Partitioning

Because of data volume, a point cloud web service must support query patterns that allow clients to limit the amount of data returned by both 2D bounding box or a 3D cube with both being controlled by tree depth. The ability to query with oriented frustums could also be nice to have. Without these abilities, applications are doomed to over-select data with a significant network and processing performance penalty.

6.1.4. Attribute Filtering

In combination with spatial range queries, the ability for client applications to make range queries on any of the point cloud’s attribute information is also valuable. This capability allows clients to control their over-selection of data and offload data filtering to a server-side resource.

In situations where a server is responding to client traffic, it can simply filter data before responding. In a static tile-based data approach, the data must be organized to allow the client to make individual requests for desired attributes. This round trip time (RTT) must be balanced with the over-selection required of simply fetching the extra data and dropping it on the floor. Active servers have an advantage on this requirement, due to the fact that they can individually respond to each request with computing resources.

6.1.5. Alternate Representations

A common first task of point cloud data consumers is the rasterization of their data. In addition to being able to provide point cloud data without spatial our attribution quantization, simplification can allow bulk reduction of the data with situationally minimal information loss. A point cloud web service implementation that can respond with data that represent rasterized or voxelized data without extra server-side processing can provide significant convenience to downstream client applications.

6.1.6. Level of Detail

Server-side data resampling or subsampling is a frequently desired feature by clients of point cloud web services. The ability for a client to fetch progressively more or less dense representations of the data without additional processing is a hard requirement for applications that wish to provide live preview and visualization of a point cloud web service over a network in order to achieve adequate performance.

Clients must be able to control their access to the level of detail without repeating requests or making extra requests that are expected to fail to signify data. The ability to look ahead prevents clients from making many wasteful extra round trips to the server that simply fail due to the way that point cloud data are organized.

6.1.7. Format Compatibility

ASPRS LAS is a dominant interoperability mechanism in geospatial point clouds, and any point cloud web service approach must be able to provide compatibility with prevalent formats such as LAS. Capabilities required to achieve that requirement include supporting flexible data schemas, providing extensible and standard metadata storage, and building upon existing scope-limited standards such as version 2 of the Well-Known Text (WKT) (OGC 12-063r5) when applicable.

6.1.8. Compression

Point cloud data have challenging size, and a compelling compression story is probably a required feature of any point cloud web services stack. Ideally, the compression should be widely supported across multiple computing platforms (Native, JavaScript, Java), efficient in both space and time dimensions (tilted toward decompression), and have no intellectual property constraints that prevent implementation in a multitude of industries.

Lossy encodings can have their place, but for most situations, especially in archival usage scenarios, lossless content encodings are paramount.

6.1.9. Metadata Organization

The metadata a point cloud web service provides must satisfy two specific needs – support information to allow clients to query through the resource and auxiliary metadata of any composition that clients can attach to resources, sub-resources, tree nodes, or compositions of data.

The support metadata should be organized in a convenient-to-consume fashion, with human-readable content desirable (but not required). Redundant metadata should be avoided, especially for information that defines resource composition and structure.

6.2. Optional

A checklist of every requirement may not be achievable for a point cloud web service. Depending on client requirements, some design choices may be in opposition in such a way as to not allow them to both be implemented. The following section discusses a list of "Optional" requirements that represent a list of useful features that point cloud web services should strive to possess. If these options were available, they could certainly help with the utility and capability of web services focused on point clouds.

6.2.1. Surface Construction

In addition to alternative quantized representations like voxels, 3D surface constructions such as Triangulated Irregular Networks (TINs) and meshes are a common secondary derived product that point cloud consumers often utilize. A web service that makes it convenient to construct or provides these surfaces has the potential to be advantageous by offloading work each client must often do to the server. Many times, these data are pre-computed to a specific resolution and specification, and it is quite difficult to achieve on-the-fly operation of surface construction algorithms in a server application with satisfactory performance.

6.2.2. Decentralized Redundancy

Point cloud web service data organizations that allow for idempotent responses without significant server intervention also allow for addressable redundant data storage. In a peering data network scenario, this property has the advantage of allowing partial data homing to increase redundancy and potentially improve responsiveness. Peering protocols such as Dat [https://datproject.org/] support distributed synchronization, revision changes, and encryption which may be attractive to organizations looking to provide authoritative point cloud data services.

Idempotent requests are also an important consideration that can interact with data organization. Multiple requests for the same resource should always return the same results. This indemnity allows servers to treat data atomically, and it allows clients to parallelize simply. All of the web services discussed in the Contrast section provide idempotent access to point cloud data.

6.2.3. Temporal Partitioning

Data selectivity based on time or other attributes should be made possible by a point cloud web service approach. While the primary selectivity constraint is likely to be along spatial partition, other attributes such as collection time, metadata, or point attribute features are desirable. Without this selectivity, client applications are likely to over-select data, resulting in extra download cost and client processing time.

Chapter 7. Web Service Implementation Contrast

In this section, we discuss five different point cloud web service approaches and contrast them on the requirements described in Chapter 6. The requirements are segmented into **Required** and **Optional**, with the bulk of the discussion on the Required topic. A short summary table in Section 7.6 distills the rather large topic into a more easily digestible form.

Five point cloud web service approaches are compared, including two OGC Community Standards. The point cloud web service software tools (Esri I3S [https://github.com/esri/pointcloud], 3D Tiles [https://github.com/AnalyticalGraphicsInc/3d-tiles], Greyhound [https://greyhound.io], PotreeConverter [https://github.com/potree/PotreeConverter], and Entwine [https://entwine.io]) are contrasted. Data access (read support) to all five implementations is provided by open source software, and software to create (write support) point cloud web services is provided by four. Three of the services, PotreeConverter, Greyhound, and Entwine, evolved together toward the combined problem of providing high performance for browser-based rendering scenarios. PotreeConverter innovated the "hierarchy organization" approach to metadata that allows clients to look ahead before querying data, and this feature is also reflected in Greyhound and Entwine. Greyhound, Entwine, and PotreeConverter all utilize the industry-accepted LASzip as its compressed content encoding, while Esri I3S innovates its own in the form of LEPCC compression and 3D Tiles leverages browser-capable encodings.

7.1. Esri I3S

Esri I3S, is available as an OGC Community standard, but the initial specification was not complete for point clouds. In February 2018, Esri released documentation for the point cloud portion I3S specification on GitHub at https://github.com/Esri/pointcloud [https://github.com/Esri/pointcloud].

The point cloud portion of I3S represents a static tile set with data organized in separate files by attribute type. It is expected to be consumed by web clients over the internet, it is suitable to feed a browser-based renderer or data processing software, and it can be configured to store data in a lossless configuration. A complimentary package encoding called SLPK is also used to store I3S data in a form suitable for single-file disk storage or archival network transit.

7.1.1. Required Requirements

Flexible Schema

I3S allows for storage of any attribute type, with three specific encodings allowed – `embedded-elevation`, `lepcc-intensity`, and `lepcc-rgb`. Some attributes are aggregated, such as XYZ, number of returns / return number, and RGB, while others such as intensity are stored in their own file for each data node. In a worst case scenario, which most data rendering activities would not encounter, a full fetch to aggregate data for a single tree node may require a request for each attribute minus potentially six combined attributes for items such as "Y", "Z", "Green", "Blue", "Return_num", and "Return_count". Each individual round trip time (RTT) via HTTP could possibly have a fixed cost of 20-40 milliseconds depending on network conditions in addition to any decompression time. Multi-threaded fetches can aggregate the i/o and reduce the total time, but systems must be in place to handle the request load. The RTT consideration dissolves in the package-based SLPK scenario, however.

Configurable Data Sources

I3S allows for data services to be configured as desired, merged on the client as needed, and be stored in various coordinate systems as required.

Spatial Partitioning

I3S clients can limit their queries to specific data by bounding volume and resolution when data are selected in cooperation with tree metadata.

Alternative Representations

I3S clients can provide data for a given density, which could be realized as a sparse, voxelized representation of the source data. No alternative representations are stored directly with the point data, but the I3S specification allows for other data types, such as meshes, rasters, and textures, to be coincidentally stored with the point cloud data.

Level of Detail

I3S data is stored with replacement refinement, and points are duplicated or overviewed at different gridded resolutions throughout the tree – much like is commonly found in raster data organization. Original source coordinates from the input data input source are stored in the base leaf nodes for a lossless data organization if compression settings are specified. This organization can add a small amount of storage overhead for the lower resolution overviews. The ratio of duplicate data refetching is calculable and configurable at data organization time, and the total sum of the overview cost is only felt during a full tree traversal, which is an uncommon scenario.

Format Compatibility

The only known software to create I3S point cloud web services at the time of writing is ArcGIS from Esri. Support for input formats is limited to the list provided by that platform.

Compression

LEPCC [https://github.com/PDAL/lepcc/tree/master/src] is an adaptation of Esri's LERC [https://github.com/esri/lerc] encoding used in raster situations, and it allows the user to specify a point cloud quantization level and compress data to an explicit and expected precision loss. Most LiDAR and point cloud data does not require more than three decimals of precision, especially in aerial acquisition scenarios, yet floating point encoding of point clouds introduces significant bit weight to the data for full preservation. Other attributes are stored with generalized compression techniques such as Gzip [https://en.wikipedia.org/wiki/Gzip], which has widely available software implementations many computing platforms.

NOTE Esri has a more complete description of MRF online [https://community.esri.com/servlet/JiveServlet/downloadBody/7473-102-1-9153/MRF_CloudOptimizedImageFormat_LERC_Compression.pdf].

Metadata Organization

I3S hierarchy organization is split linearly and optimized for accessing nodes near each other spatially at the same level of detail, which is a typical query pattern associated with rendering. Spatial metadata is conveyed in Earth-Centered Earth-Fixed (ECEF) with quaternion rotation, and clients must be able to reproject from the source spatial reference to ECEF coordinate system to decode oriented bounding box information.

To determine leaf nodes for data reconstruction, the entire depth of the hierarchy metadata must be traversed and the upper levels discarded to determine the depth at which each leaf node resides.

7.1.2. Optional Requirements

Surface Construction

I3S supports storage of meshes, raster data, and textures, but no on-demand surface construction can be made for data without purpose-specific software.

Decentralized Redundancy

I3S data are compatible with a decentralized redundant storage network system.

Temporal Partitioning

I3S data resources must be temporally organized to allow for explicit selection and partitioning, and the I3S specification allows for such organization approaches.

7.2. PotreeConverter

PotreeConverter is an open source software project from Markus Schütz [https://www.researchgate.net/profile/Markus_Schuetz2] originally of Technische Universität Wien (TU Wien) that was developed in support of the Potree browser-based point cloud renderer. It is available online at PotreeConverter [https://github.com/potree/PotreeConverter]. PotreeConverter innovated a number of features in support of browser-based point cloud rendering. These include the notion of a pre-built, static hierarchy that defines the shape of the tree, storage of LASzip data objects that were friendly to cloud object storage scenarios, and a point data structure that conveniently maps to rendering capabilities.

7.2.1. Required Requirements

Flexible Schema

PotreeConverter only supports storage of LAZ with a fixed schema. Additionally, PotreeConverter does not provide full support for the entire LAS content schema, and only a selected subset is provided.

Configurable Data Sources

PotreeConverter allows for data services to be configured as desired, merged on the client as needed, and be stored in various coordinate systems (cartesian) as required.

Spatial Partitioning

Data is accessed using a fixed tiling query pattern, and clients must project and construct data requests to fetch data at expected windows and depths. Clients can optionally use the static hierarchy information PotreeConverter provides to adjust the pace and batching of the queries. Potree clients can limit their queries to specific data by bounding volume and depth when data are selected in cooperation with tree metadata.

PotreeConverter's tree structure is a tree of binary files describing the density and presence of data. More information about PotreeConverter tree structure can be found at https://github.com/potree/potree/blob/develop/docs/potree-file-format.md#index-files [https://github.com/potree/potree/blob/develop/docs/potree-file-format.md#index-files].

Alternative Representations

No alternative data representations other than points are stored in PotreeConverter converter output.

Level of Detail

PotreeConverter data is stored with additive refinement, and a full view of points are accumulated as clients select through the tree to higher depths. Original source coordinates from input data are stored with an explicit scale and offset, similar to all ASPRS LAS and LAZ data.

Format Compatibility

PotreeConverter uses libLAS [https://liblas.org] for LAS and LASzip read support, and it also includes capability to read PLY, XYZ, and PTX files. PotreeConverter does not store the complete data attribute list from LAS or LAZ data, however, and some attributes such as GPSTime, return information, and UserData is removed from the output. This feature makes PotreeConverter unsuitable in data warehousing scenarios if the output is expected to be the only data organization available.

Compression

PotreeConverter exclusively uses LASzip for data compression. LAZ 1.0 to 1.3 is supported by Potree, with LAZ 1.4 expected in a future revision.

Metadata Organization

PotreeConverter implements the data organization steps for Potree [http://potree.org], and it makes a number of choices to optimize that application's usage of the data. First, the data are organized as an octree, and the metadata (hierarchy) storage of the tree splits along predictable and computable boundaries to eliminate the need for storage and response of redundant data. Second, the metadata tree stores point count information per-node to allow client applications to collect and estimate the response size of requests that are implicitly defined by the structure of the tree. Lastly, the metadata stores shape-of-tree information and support metadata such as defined attributes.

Without the hierarchy information that PotreeConverter constructs, querying applications would be required to make numerous requests for data that were doomed to fail, and the application would have to track those failures as indication variables that define the shape of the data stored in the tree.

NOTE More information about PotreeConverter's metadata can be found on GitHub [https://github.com/potree/potree/blob/develop/docs/potree-file-format.md#meta-information].

7.2.2. Optional Requirements

Surface Construction

Only point data is stored in PotreeConverter output.

Decentralized Redundancy

PotreeConverter data are compatible with a decentralized redundant storage network system.

Temporal Partitioning

PotreeConverter data resources must be temporally organized to allow for explicit selection and partitioning.

7.3. Greyhound

Greyhound [https://greyhound.io] is an open source point cloud web service server that allows users and developers to interact with raw point cloud data. Because it is an active server, rather than a static tile set like the rest of the approaches contrasted, it provides many convenient features at the cost of server activity, maintenance, and complexity.

Greyhound was developed with a goal of supporting browser-based online rendering of point clouds. While other activities, such as processing support and data interrogation are supported by Greyhound, its design choices were made in support of rendering activities using a REST-style HTTP interface. These choices include a query interface that allows clients to make simple idempotent requests for data using a bounding cube and tree depth.

Unlike the tile-based approaches described in this section, clients need not adapt their query structure to the data source with Greyhound. They can control the pace and attribute composition of the returned data through the Application Programming Interface (API). A corresponding consequence of this flexibility is data caching can be more difficult, data redundancy and distribution requires specialized support, and an active server must be deployed to respond to requests at all times.

7.3.1. Required Requirements

Flexible Schema

Greyhound allows for storage of data in any defined schema, with communication of the storage arrangement provided by its metadata.

Configurable Data Sources

Greyhound allows for data services to be configured as desired, merged on the client as needed, and be stored in various coordinate systems (cartesian) as required.

Spatial Partitioning

Clients can request data from Greyhound using a simple bounding cube and level-of-detail query mechanism. Queries do not need to be aligned to data indexed data boundaries, and clients can control the composition of the returned data by utilizing the `schema` parameters of their API queries. These properties of Greyhound allow clients to be very simple.

Because Greyhound queries are client-controlled, the opportunity for query caching is limited to requested resource matching. Cloud-based edge caches were shown to work well in situations where repeated requests were expected, such as a browser-based web visualization with repeated users, but a caching layer could add complexity to the overall system.

Alternative Representations

No alternative data representations other than points are stored in Greyhound converter output.

Level of Detail

Greyhound data, like Entwine/EPT data, is stored with additive refinement, and a full view of points are accumulated as clients select through the tree to higher depths. Original source coordinates from input data can be stored with an explicit scale and offset, similar to all ASPRS LAS and LAZ data.

Format Compatibility

Greyhound data can be constructed from any PDAL [https://pdal.io]-readable data source including ASPRS LAS, LASzip, BPF, PTX, PTS, PLY, and database formats such as pgpointcloud and Oracle Point Cloud.

Compression

Clients that query Greyhound can control the schema of the data returned to them. This situationally useful feature means that compression for Greyhound cannot easily follow one of the existing common point cloud data encodings such as LASzip. In support of Greyhound, "lazperf" was developed to support the compression of data using the same arithmetic encoding approach that LASzip utilizes minus its data modeling based on LAS. This approach nets a storage that is 30-50% less efficient than LAZ, but one with increased flexibility and similar compression and decompression speed performance. Additionally, lazperf also works in native and JavaScript environments for use by cloud-based data management tools and browser-based rendering clients.

In addition to lazperf, Greyhound supports an uncompressed binary format with composition controlled by the client’s request schema. Servers can apply encoding such as `application/gzip` to the data to provide a standard approach, but it is neither space- nor as time- efficient as lazperf with point cloud data.

Metadata Organization

Greyhound supports an adaptation of the PotreeConverter-style hierarchy that allows clients to peek down the tree and make resource choices such as request batching or request delays. Unlike the flexibility of Greyhound's data access API, hierarchy queries must use a specific query pattern that follows the implicit shape of the tree defined by the resource. Metadata is provided to API consumers to allow them to make appropriate requests to fetch this information, and a short protocol is defined that constructs the implicit shape of the data organization to allow it.

7.3.2. Optional Requirements

Surface Construction

Only point data is stored in Greyhound output. No textures, meshes, or raster information can be stored or served by Greyhound.

Decentralized Redundancy

Greyhound source data are compatible with a decentralized redundant storage network system, but a specialized server would need to be employed to take advantage and serve data utilizing such a network.

Temporal Partitioning

Greyhound data resources must be temporally organized to allow for explicit selection and partitioning.

7.4. Entwine

The Entwine Point Tiles (EPT) format is a formalization of the Entwine data organization approach which was constructed in support of Greyhound. EPT removed the server requirement of Greyhound in exchange for mandating that clients query against a fixed data structure. The fixed structure is more convenient for data management scenarios, and requires simpler infrastructure.

7.4.1. Required Requirements

Flexible Schema

EPT allows for storage of data in any defined schema, with communication of the storage arrangement provided by its metadata. Entwine also allows storage of a flexible list of dimensions, even when using the LASzip encoding, by storing additional dimensions as "extra bytes" fields in the LAS files. This allows applications to skip data they do not understand, while more capable clients can consume data with full fidelity.

Configurable Data Sources

EPT allows for data services to be configured as desired, merged on the client as needed, and be stored in various coordinate systems (cartesian) as required. Users can create resources with Entwine using whichever organizing principle makes the most sense. For some situations, an entire data capture over a long time period, such as the acquisition of LiDAR for an entire country, might make the most sense. In others, specific acquisitions for specific dates might be collated as needed from individual services. Each approach is possible with EPT.

Spatial Partitioning

Data is accessed using a fixed tiling query pattern modeled on the concepts of a WMTS [http://www.opengeospatial.org/standards/wmts] or TMS [https://en.wikipedia.org/wiki/Tile_Map_Service] raster tiling scheme with the addition of tree depth as another parameter. The data organization is oriented toward web developers with familiarity with WMTS- or TMS-style data access patterns. A full description of how to access data in EPT is provided on GitHub [https://github.com/connormanning/entwine/blob/master/doc/entwine-point-tile.md].

Alternative Representations

No alternative data representations other than points are stored in EPT. An EPT tree is realized as a cubic domain, and applications can project points into voxel or raster cells as needed to create a rasterized or voxelized representation of data.

Level of Detail

Entwine organizes the data as an octree, and clients are expected to accumulate views of data by interpreting a deterministic tree shape from the resource’s metadata and utilizing EPT’s "hierarchy" metadata to provide look-ahead metadata for applications querying up and down the tree. EPT data is stored with additive refinement, and a full view of points are accumulated as clients select through the tree to higher depths. Original source coordinates from input data can be stored with an explicit scale and offset, similar to all ASPRS LAS and LAZ data.

EPT borrows the hierarchy metadata approach that PotreeConverter implemented to free itself from having to choose between a space-organized or a density-organized approach. The hierarchy allows clients "peek" down the octree from a node position to provide an exact count of upcoming points that would be selected. This information allows clients to adjust their pacing while batching their round trips appropriately. Without this mechanism, clients are at the mercy of their responses, and they have little control over how long they must wait to complete a response cycle.

Format Compatibility

EPT data can be constructed from any PDAL [https://pdal.io]-readable data source including ASPRS LAS, LASzip, BPF, PTX, PTS, PLY, and database formats such as pgpointcloud and Oracle Point Cloud.

Support of browser-based rendering clients was a primary goal of EPT, just as it is for all of the other point cloud web service approaches on this list, but explicit support for archival data management was also a requirement. Because the data collections are typically so large, a web service that does not support a complete reconstruction of source information has the effect of duplicating the storage and management cost of the data. For collections that could be petabytes in size, this property is quite expensive. EPT was designed to support storage and metadata of point cloud data to completely reconstruct original data. This feature allows it to be conveniently utilized as the only data storage necessary in data warehousing situations.

Compression

In addition to an uncompressed binary encoding, Entwine EPT supports LASzip as the data encoding for data storage, and any attribute types not explicitly defined by the LAS format are stored as "extra bytes" dimensions on the data. This organization has three specific benefits:

- LAZ is widely supported by the LiDAR and point cloud industry
- LAZ data based on ASPRS LAS 1.4 can store any attribute type with the exception that readers must understand them.
- LAZ is supported by browser-based JavaScript and native C++ open source software implementations.

Alternative encodings could be supported by EPT so long as support for flexibly-defined schemas was possible. No alternative content encodings are specified beyond LAZ and uncompressed binary are specified at the time of writing, however.

Metadata Organization

Entwine supports a PotreeConverter-style hierarchy metadata mechanism that allows clients to peek ahead of their queries to batch requests and plan resources ahead of capturing the results of responses.

7.4.2. Optional Requirements

Surface Construction

Only point data is stored in EPT output. Entwine leaves it to clients to provide surface construction capabilities, but it is noted that octrees are often constructed as a data indexing approach for 3D surface construction algorithms, and clients can utilize the data organization provided by EPT to achieve these surfaces.

Decentralized Redundancy

EPT source data are compatible with a decentralized redundant storage network system, and the additive refinement storage approach provided by it helps to eliminate total system storage overhead in redundancy scenarios.

Temporal Partitioning

EPT does not provide explicit support for temporal or alternative dimensional range indexing. Data resources must be specifically organized by time to support secondary indexing.

7.5. 3D Tiles

3D Tiles is a specification from Analytical Geographics Inc. currently incubating in the OGC Community Standard process as of fall 2018. The current specification can be found at https://github.com/AnalyticalGraphicsInc/3d-tiles [https://github.com/AnalyticalGraphicsInc/3d-tiles], and it is expected to be available via the OGC Community Process sometime in 2019. Like I3S, 3D Tiles contains a data structure organized in support of streaming and rendering 3D geospatial content, and it includes support for textures, solids, triangles, and point clouds. In this section we will contrast the point cloud support of 3D Tiles and describe how it achieves support of properties required of point cloud web services.

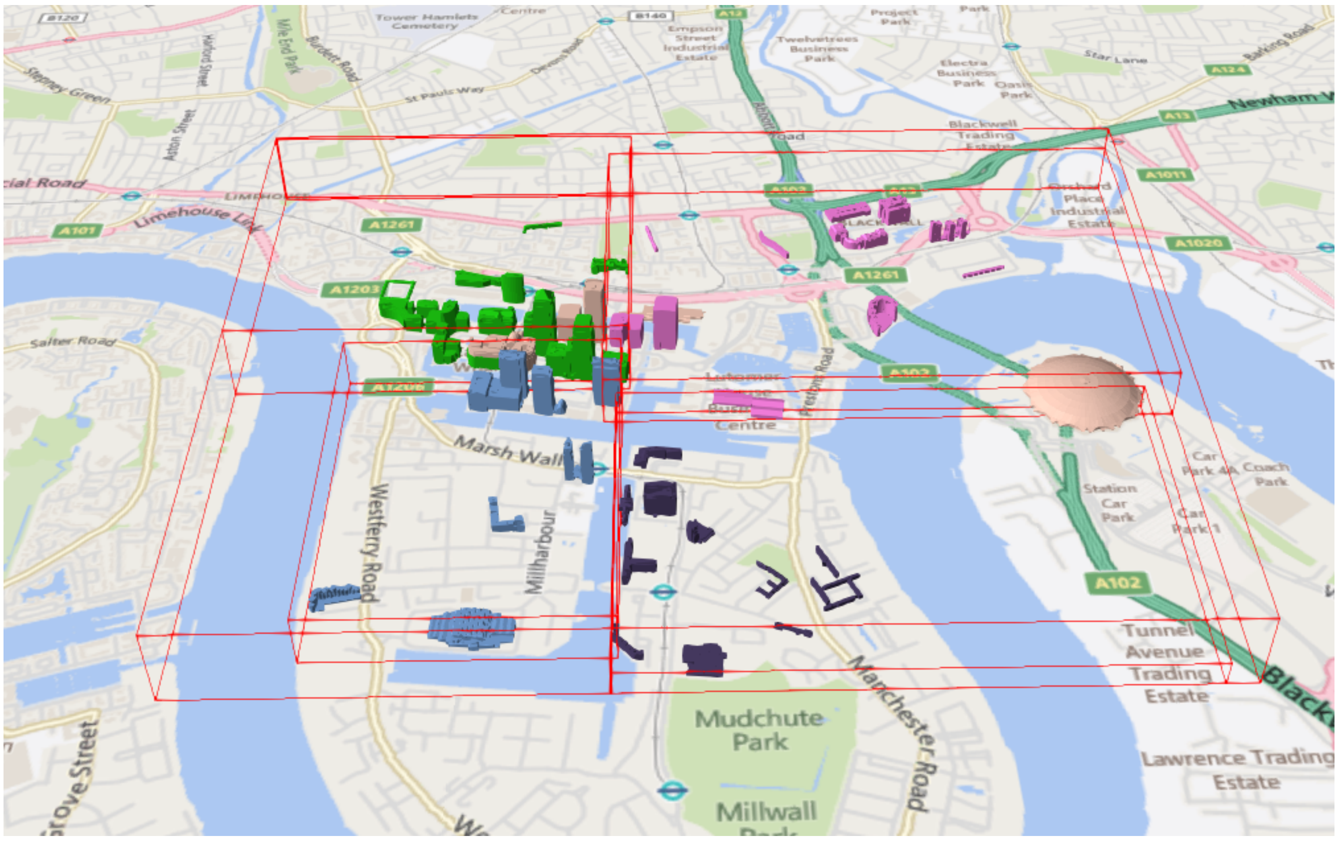


Figure 2. 3D Tiles allows encapsulation of many tree variations such as loose quadtree or octrees at the expense of additional metadata for implicitly defined data structures. Figure from *https://github.com/AnalyticalGraphicsInc/3d-tiles [https://github.com/AnalyticalGraphicsInc/3d-tiles/blob/master/specification/figures/nonUniformQuadtree.png]*

7.5.1. Required Requirements

Flexible Schema

First-class point cloud data attributes are limited to position, color, and normal information in 3D Tiles. Other attributes may be stored, and are are advertised at the per-data-node level rather than the resource level.

3D Tiles provides an excellent wire format that is pre-packed for insertion into GPU rendering pipelines, with spatial coordinates be expressed as 32-bit floating point plus 32-bit origin or 16-bit scaled plus offset integers.

Configurable Data Sources

3D Tiles allows for data services to be configured as desired, merged on the client as needed, and be stored in various coordinate systems as required.

Spatial Partitioning

3D Tiles is focused on convenient data organization structures in support of online data rendering, and features of the format reflect choices in support of those goals. It chooses to be highly flexible for data providers, and it was not explicitly designed with data archive as a goal. A specific consequence of this perspective is the requirement for explicit metadata about the shape and organization of the data tree to be provided.

Alternative Representations

No alternative representations are stored directly with the point data, but the 3D Tiles specification allows for other data types, such as meshes, rasters, and textures, to be coincidentally stored with the point cloud data.

Level of Detail

3D Tiles provides a very flexible level of detail structure that clients can leverage to support fast selection of data in support of activities such as rendering.

Format Compatibility

3D Tiles data can be constructed from any PDAL [https://pdal.io]-readable data source including ASPRS LAS, LASzip, BPF, PTX, PTS, PLY, and database formats such as pgpointcloud and Oracle Point Cloud. A conversion from EPT using the Entwine software toolchain provides one open source software path to writing 3D Tiles point cloud data. Other conversion software include https://github.com/tum-gis/cesium-point-cloud-generator [https://github.com/tum-gis/cesium-point-cloud-generator] and https://github.com/commaai/entium [https://github.com/commaai/entium].

Compression

While the OGC Community Standard candidate does not specify a compressed point cloud encoding, the latest version of the specification does specify using Draco [https://github.com/google/draco]. Browser-supported content encodings, such as `application/gzip` are also supported.

Metadata Organization

The metadata organization for 3D Tiles is very flexible, which can make it verbose for implicit quadtree or octree structures conveniently suited for point clouds. Bounding volumes can be expressed as rectangular volumes, spheres, or oriented bounding boxes in 3D Tiles.

7.5.2. Optional Requirements

Surface Construction

3D Tiles supports storage of meshes, raster data, and textures. Because 3D Tiles has no live server, no on-demand surface construction can be made for data. Storage of a surface construction output in 3D Tiles is fully supported.

Decentralized Redundancy

3D Tiles source data are compatible with a decentralized redundant storage network system.

Temporal Partitioning

3D Tiles data resources must be temporally organized to allow for explicit selection and partitioning, and 3D Tiles supports sophisticated temporal description and partitioning capabilities.

7.6. Requirements Matrix

We define a simple categorization to allow some gradation in defining a particular software suite's support for a particular requirement. A value of **0** states no support for the requirement and essentially no work-arounds are possible. A value of **3** states that full support for the requirement is available in all software configurations.

Table 3. Support categories

Value	Description
0	No support possible
1	Partial support with limitations
2	Support in most configurations
3	Support in all configurations

This section defines a simple categorization to allow some gradation in defining a particular software suite's support for a particular requirement. A value of **0** states no support for the requirement and essentially no work-arounds are possible. A value of **3** states that full support for the requirement is available in all software configurations.

Table 4. Required features comparison

	I3S	Potree	Greyhound	EPT	3D Tiles
Flexible Schema	2[a]	0[b]	3	3	1[a]
Configurable Data Sources	2	1[c]	3	3	2
Spatial Partitioning	3	3	3	3	3
Attribute Filtering	3	0	3	1[d]	1[d]
Alternate Representations	2[e]	1	1	1	2[e]
Level of Detail	3	3	3	3	2[f]
Format Compatibility	2	2	3	3	2
Compression	3	3	2	3	2

Table 5. Required Requirements Footnotes

ID	Footnote
a	Schema has both a fixed and a flexible component. Not all dimensions of the fixed component exist in all configurations.
b	Schema is completely fixed, although some attributes may not be present
c	Only ASPRS LAS and LASzip formats are supported for ingestion.

ID	Footnote
d	Attributes are combined and cannot be selected individually.
e	Data structure supports storage of alternative representations such as meshes and rasterized products, but no products are produced in response to queries.
f	Access to level of detail requires selecting metadata to support queries rather than implicit data selection based on computed depth/tree position.

Table 6. Optional features comparison

	I3S	Potree	Greyhound	EPT	3D Tiles
Surface Construction	2	0	0	0	2
Decentralized Redundancy	3	3	0	3	3
Temporal Partitioning	1	1	3	1	1

Chapter 8. Discussion

The investigation described in Chapter 7 was upon software and services that are still in the maturation phase of their lifecycle. Each software stack evolved toward a framework that could act in service of live-preview web-based renderers, along with features that tilt the services toward specific application uses.

None of the point cloud web service approaches described in Chapter 7 met all of the requirements described in Chapter 6, which is an unsurprising result given that each was developed with a specific (and sometimes different) application in mind. For web service specifications coming from OGC, the following additional requirements must be considered:

- Static tiling approach aligned with WMTS and GeoPackage.
- Suitability for archival data storage and transmission in addition to progressive access over HTTP network protocols.
- Downstream application performance sensitivity to data organization.

These additional items along with the requirements described in Chapter 6 would be best considered in a future OGC Testbed.

8.1. Web Service Outlook

The following section concentrates on two specific topics for the web services contrasted in Chapter 7 – content encoding and tree hierarchy. These two topics are critical to a static point cloud web service and future testbed activities should focus on describing the need and arrangement of these two features for any point cloud web service that OGC seeks to standardize.

8.1.1. I3S/SLPK

The I3S/SLPK specification provides excellent coverage of many items on the requirements list, but the current linear expression of its tree metadata structure can induce additional RTTs that applications must take before selecting data. Because each RTT typically has a fixed time cost and the information is needed in sequence, the cost of this choice can be significant for some application scenarios.

A challenge for some clients with I3S point clouds organization are fixed depths of the tree do not correspond to fixed resolutions, which is contrary to expected octree behavior. This does not give clients a deterministic change in resolution as they traverse through the tree. Attributes are also organized in I3S in such a way that can require per-attribute RTT access. This property can impose a significant request cost overhead, especially in archive or full-data transformation operations with each RTT having a fixed time cost component.

8.1.2. PotreeConverter

PotreeConverter implemented a pioneering implementation of a static point cloud web service, and features such as hierarchy (discussed in Chapter 7) from that implementation have spread to other service such as Greyhound and Entwine. PotreeConverter is set to be quickly superseded by

Entwine/EPT. EPT represents an evolution of PotreeConverter output to overcome some of its significant deficiencies – notably the inability to store a flexible schema, a limited metadata story, and a sparse supported format list.

8.1.3. Greyhound

Greyhound is an active server approach to a point cloud web service. This includes being able to cover convenient features such as Attribute Filtering in exchange for the overhead of maintaining an active live server. For many simple rendering situations, this overhead is often not worth the performance or capabilities trade-off.

Greyhound does have a niche in situations where clients want to be extremely simple (such as querying only by a bounding cube and depth), or in situations where clients look to have a remote server that is living close to the bulk of the data do something to it on the way to the client.

8.1.4. Entwine

Entwine Point Tiles (EPT) is an evolution of the implicit additive static tile approach of PotreeConverter with the metadata, access, and content encoding of Greyhound. While it does not provide full coverage of the requirements list, it does a good job of meeting most of them while balancing a number of challenging properties that are in opposition. As described in Section 7.4, it is a specification that takes the experience learned in development of PotreeConverter and Greyhound and applies it to static tiling web services for point clouds.

Open source software provides both read and write support, the approach has been demonstrated on some very large data services Entwine mitigates the RTT overhead of approaches such as I3S and softens the metadata overhead of a fully dynamic approach like 3D Tiles. Finally, while it currently utilizes the LASzip content encoding widely used throughout the industry for archival data storage, the access pattern and metadata approach of EPT could be applied to any content encoding.

8.1.5. 3D Tiles

3D Tiles allows expression of both hierarchical tree structures provided by I3S and PotreeConverter/Entwine, but it currently lacks an implicit additive tree structure as part of the specification. Developer discussion to provide support for one is underway.

3D Tiles has adopted the Draco compression engine from Google for point clouds in recent versions of the specification. Draco does not yet have wide adoption in many genres that provide point cloud data such as the LiDAR and satellite processing industries, but if it is shown to be well suited, adoption should grow.

Chapter 9. Conclusion

The investigation described in Chapter 7 was upon software and services that are still in the maturation phase of their life cycle. Each software stack evolved toward a data organization framework that could act in service of live-preview web-based renderers, along with features that tilt the services toward specific application uses.

None of the point cloud web service approaches described in Chapter 7 met all of the requirements described in Chapter 6, which is an unsurprising result given that each was developed with a specific (and often different) application in mind. For web service specifications coming from OGC, the following additional requirements must be considered:

- Static tiling approach aligned with WMTS and GeoPackage.
- Suitability for archival data storage and transmission in addition to progressive access over HTTP network protocols.
- Downstream application performance sensitivity to data organization.

Future standardization efforts should investigate how point cloud web services can fit within the rest of OGC's architecture, evaluate real-world usage scenarios with applications and servers for performance and capability, and demonstrate where existing approaches might be well suited to achieve the features and performance required.

Appendix A: Industry Survey

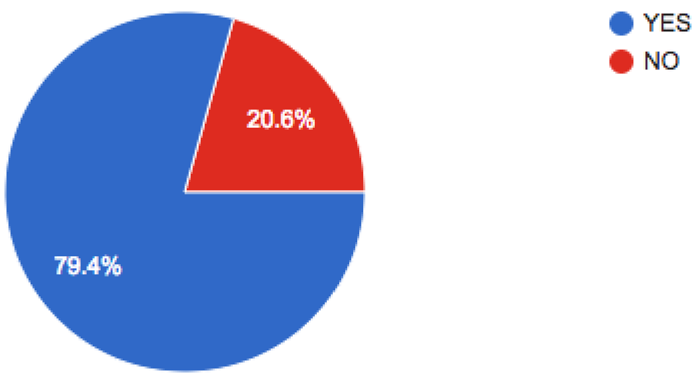
While web-based surveys are difficult to draw conclusions with due to their lack of scientific rigor, they can provide some indication and confirmation of discussion points. For this Testbed-14 Engineering Report, participants wanted to follow up on the Point Cloud Survey done by the OGC Point Cloud DWG in 2015–2016 (see the Press Release [http://www.opengeospatial.org/pressroom/pressreleases/2372] or the survey instrument [https://goo.gl/dhomVq]) with the purpose of focusing responses to specific format, protocol and niche definitions to gather the indications of the status of the market. Some of the results represent confirmation of existing biases, while others provide counter-intuitive indications of the market.

This section of the engineering report discusses the survey and its approach while speculating on some of its results.

A.1. Overview

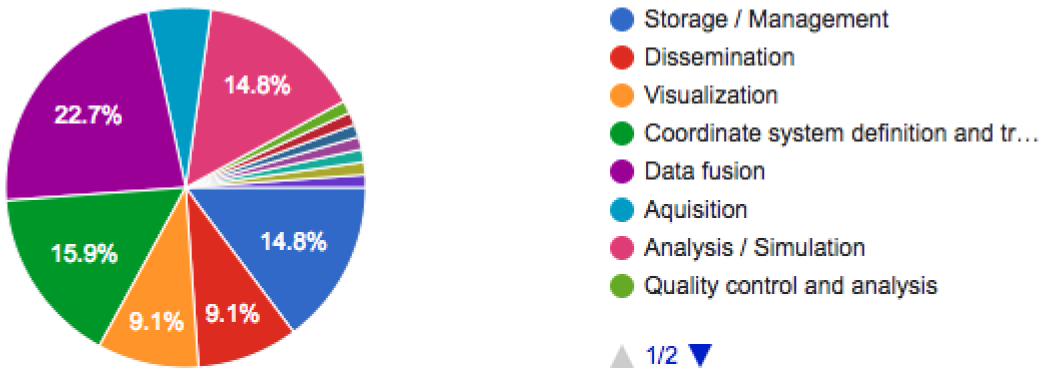
As of September 28th, 2018, the online point cloud survey had 97 respondents. The submission window was held open for about two weeks, it was advertised in the Stuttgart OGC Technical Committee (TC) meeting on September 11th, 2018, and it was circulated on a number of mailing lists and social media channels.

A.2. Interoperability Outlook



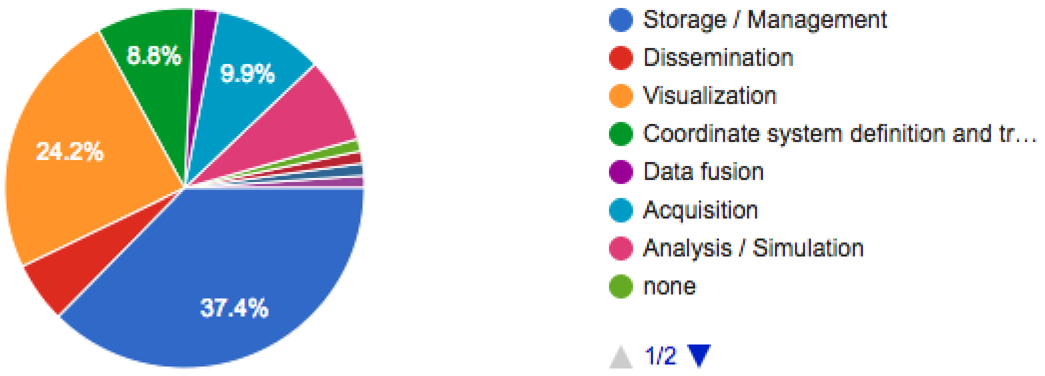
Question 1. Do you experience software interoperability challenges when working with point cloud data?

A straightforward first question was whether or not respondents even have software interoperability challenges. Curiously, 20% of respondents said they do not. We are left to wonder if this is due to completely vertical software toolchains, but it is a slightly surprising number of individuals that report not having issues.



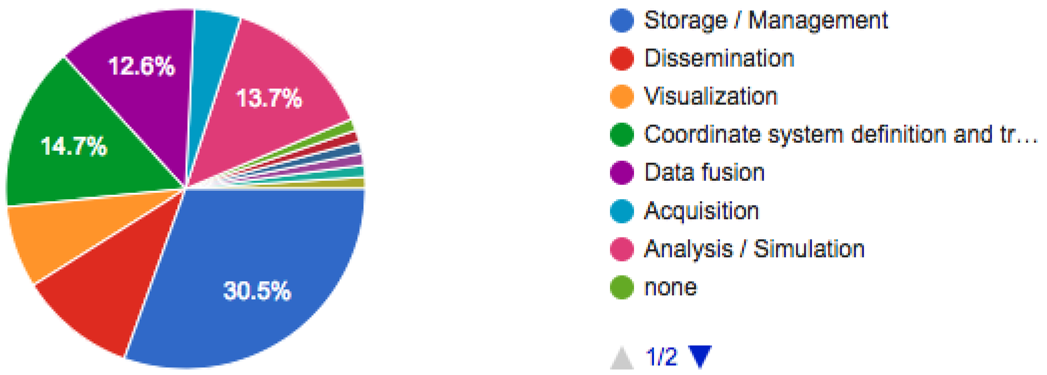
Question 2. If so, during what phase do you encounter the MOST software interoperability challenges?

Of those reporting having interoperability challenges, the topics of data fusion, analysis, coordinate systems, and storage lead rather evenly. We postulate this is likely a sign of an early maturity point cloud software industry, with tools that specialize in specific topics not always talking well with each other



Question 3. During what phase do you encounter the FEWEST software interoperability challenges?

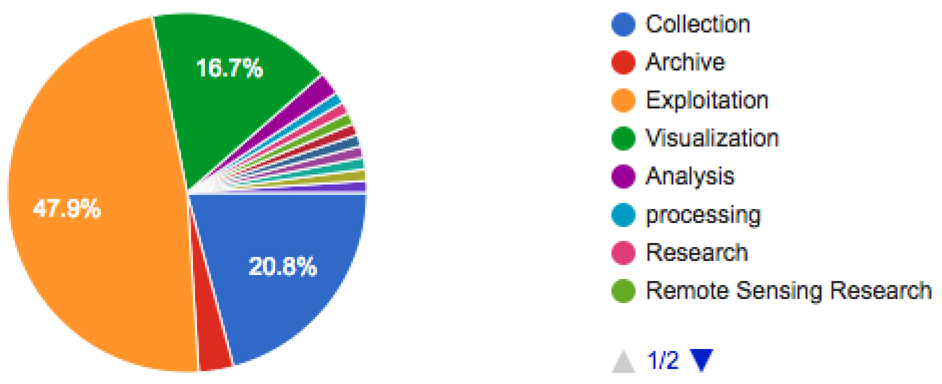
Following that up, nearly 40% of respondents stated that data storage and management represents the least challenging problem segment as far as software interoperability. The smoother interoperability provided by data storage using ASPRS LAS and LASzip, which both have wide software implementation support, may be responsible for this largest category.



Question 4. What stage is the MOST FRUSTRATING part of point cloud data workflows?

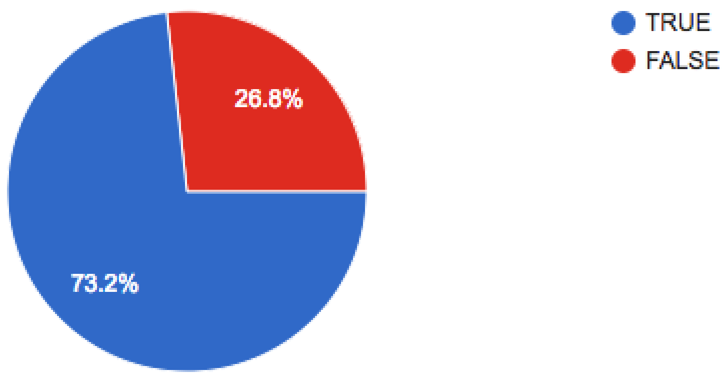
Data storage leads as a frustrating challenge with point clouds, which is to be expected given their overwhelming size and wide possibilities of storage format. While the wide availability of LAS and LAZ provides base capability for many softwares, storage and management is still a significant challenge, and the market is not satisfied with the available solutions.

Coordinate system changes interestingly rank next. Point cloud data represent measurements, and measurement frame description and coordinate system transformation is a big part of making point cloud data work well for users. That participants are complaining about the topic is an indication that more work is needed in industry software to ease the challenge for most users. Wider implementation of WKTv2 ([spec-12-063r5]) and open source software implementations provided by efforts such as GDAL https://gdalbarn.com will certainly help improve the story more widely, but specific software implementation in point cloud tools is also needed.



Question 5. What represents your primary point cloud activity?

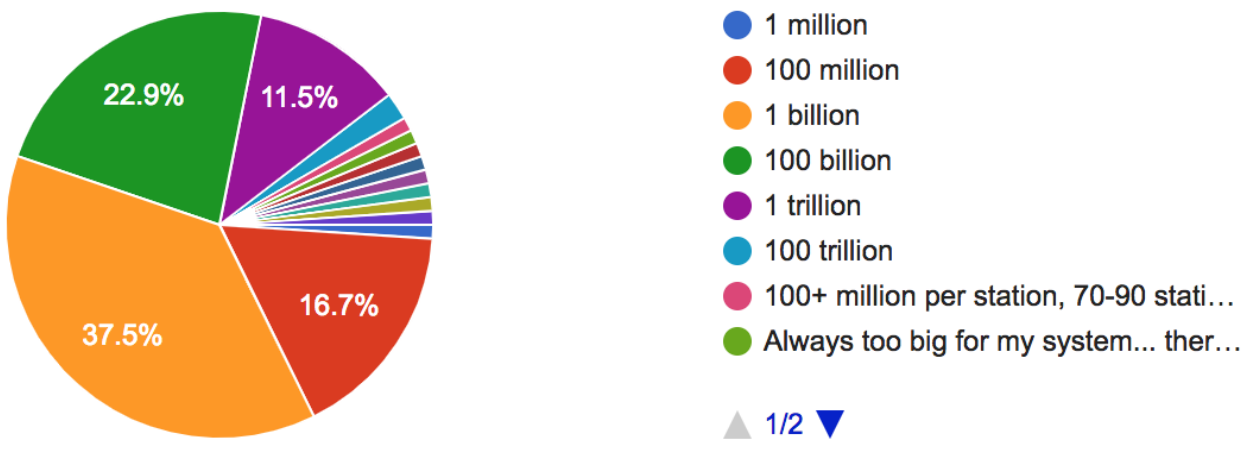
Those looking to exploit point clouds represent the largest pool of primary usage, with a wide diversity of other uses representing different tasks and segments. The response to this question highlights the need for data interoperability choices to be oriented toward exploitation needs as a primary constraint.



Question 6. I regularly work on point clouds that exceed the processing, memory, or storage capacity of my workstation or laptop.

Point cloud data volume is always described as large but exactly how large is difficult to quantify. In this question, we wondered if the description of their largess was simply whether or not they exceed the capabilities of typical workstations or laptops. Unsurprisingly the answer was overwhelmingly affirmative.

The question also is an indicator of where point cloud processing is likely to see its growth in terms of capabilities – desktop-based tools are often not up to the challenge, and dynamically scalable cloud-based computation is quickly becoming a requirement for point cloud processing scenarios.



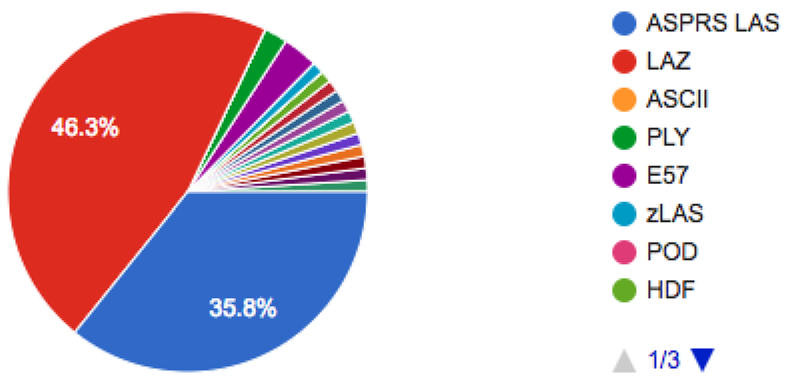
Question 7. A "big" point cloud for you has how many points?

Related to the size of point clouds people typically work with is the definition of a data collection size that is perceived as large. More than half of the respondents stated that a big point cloud was between 100 million and a billion points. This is a size at which a typical workstation would start to feel its size in terms of memory and processing, especially without additional processing to organize, filter, or subset the data.

A.3. Data Format

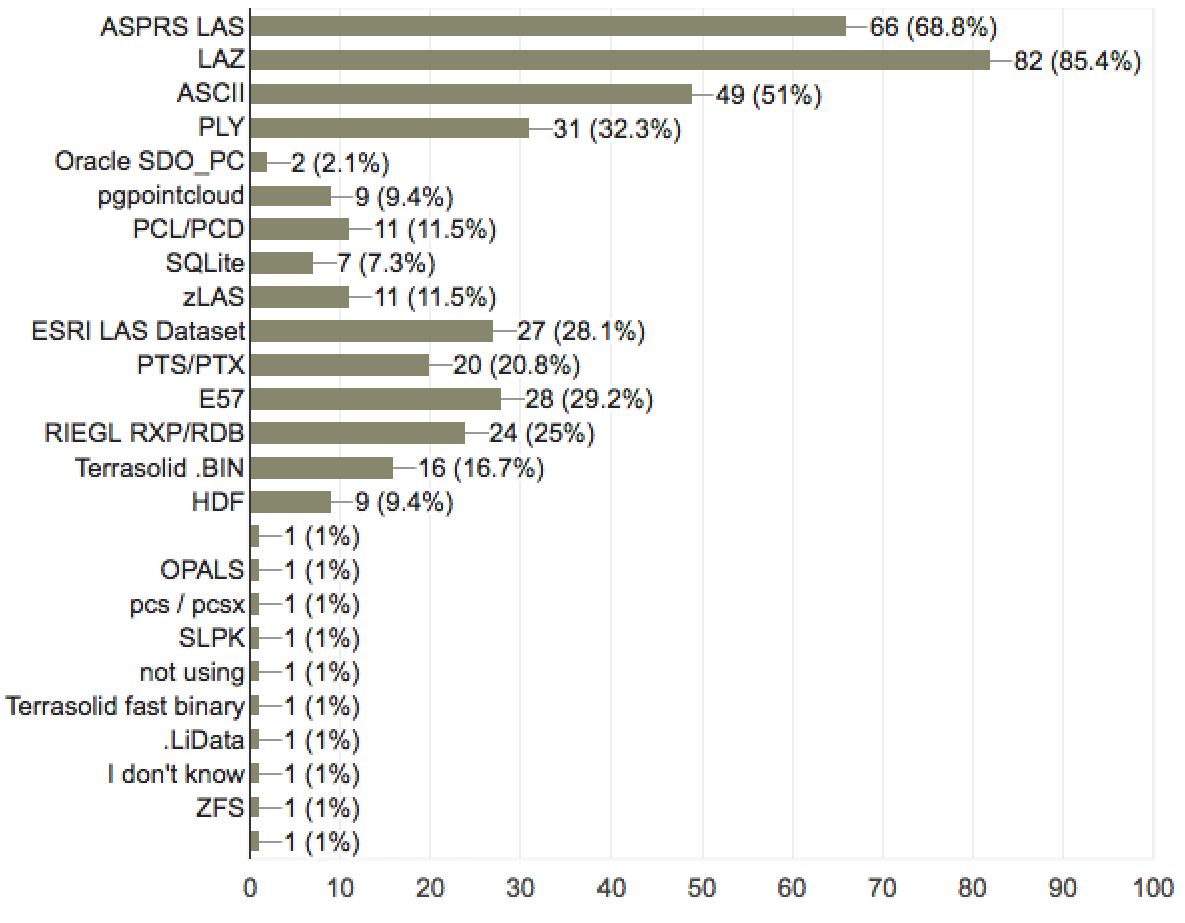
Questions about data formats for the survey mostly pertain to the most prevalent data model – ASPRS LAS – and its complimentary compression content encoding – LAZ. ASPRS LAS for geospatial point clouds as an interoperability mechanism for the storage and transmission of data has been wildly successful. Software implementations are widely available, and those implementations typically do not leak data or capability, especially for the basic features of the format.

Some missing features of LAS have become apparent as volumes of point clouds have grown and the variability of software tools have increased. Specifically, the lack of intrinsic spatial index in the format has steered applications into developing ad hoc solutions that work for specific applications, but do not work across the implementation ecosystem. Second, while LAS represents a massive storage efficiency improvement over basic text or ASCII transport of point cloud data, the LAS' lack of compression has meant that those wishing for the most widely supported interoperability story must sacrifice three to five times the data storage in contrast to a content encoding such as LASzip.



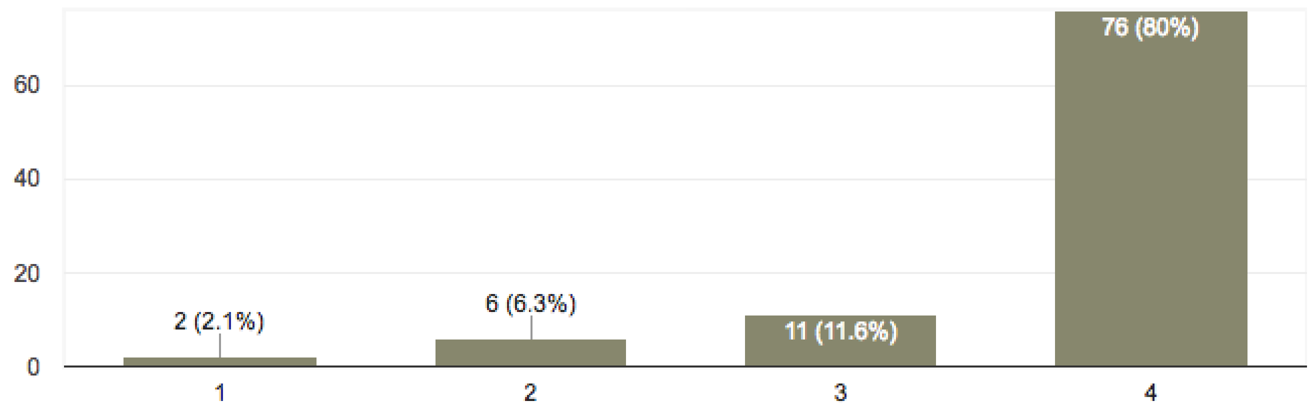
Question 8. The point cloud storage format I use the most is …

The survey confirmed the use dominance of LAS and its compressed LAZ content encoding. It is important to make the distinction that LAS is prevalent in **geospatial** point cloud domains, and other formats such as PLY are more common in non-geo situations.



Question 9. I use the following formats …

A breakdown of possibilities breaks down just how widely prevalent LAS and LAZ are. The market is missing an interoperable storage format that is suitable for archive that provides compression and spatial partitioning features. Some of the formats that respondents provide do have those features, but they are not openly specified.

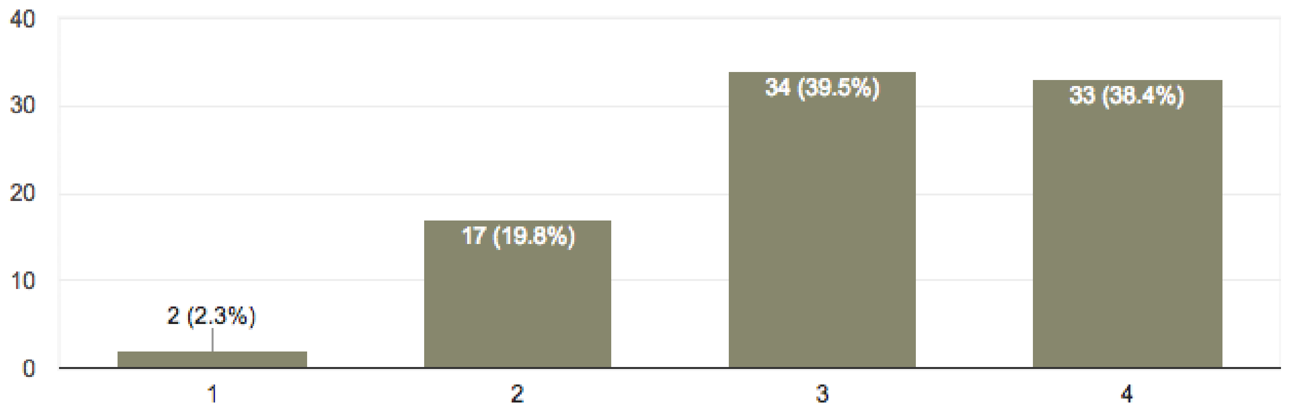


Question 10. Openly specified formats are critical to a healthy industry. (1 is disagree and 4 is agree)

Responses agree with the statement that open formats are critical, and the responses about the wide use and consumption of LAS, the most available open format for geospatial point cloud data possibly color the enthusiasm for open formats.

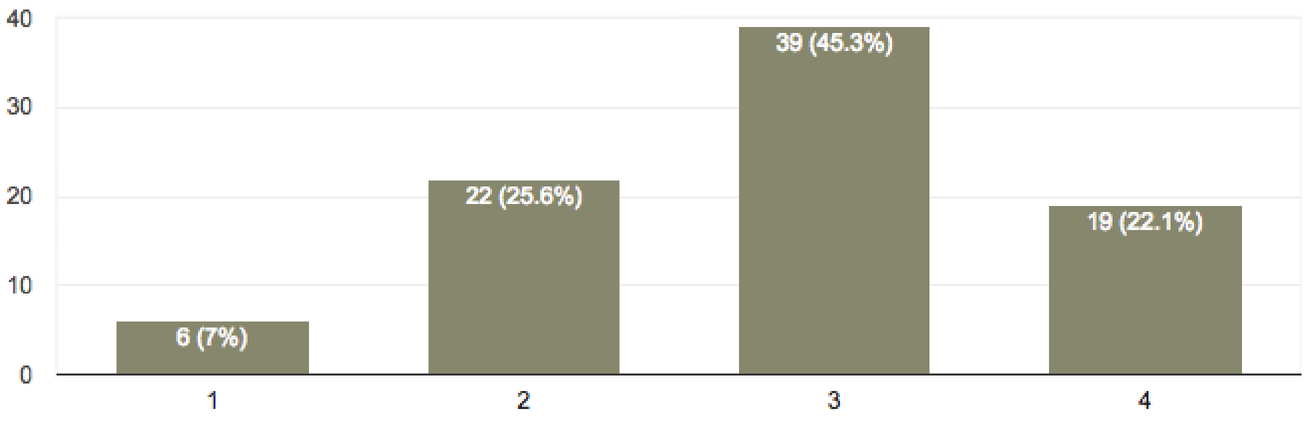
A.3.1. ASPRS LAS

Because of LAS' dominance and lack of revisions, the industry is a bit stuck in terms of interoperable abilities of its most prevalent format. A few survey questions highlight that many people are in general satisfied with LAS, but there is room for some feature growth – especially for spatial partitioning and compression.



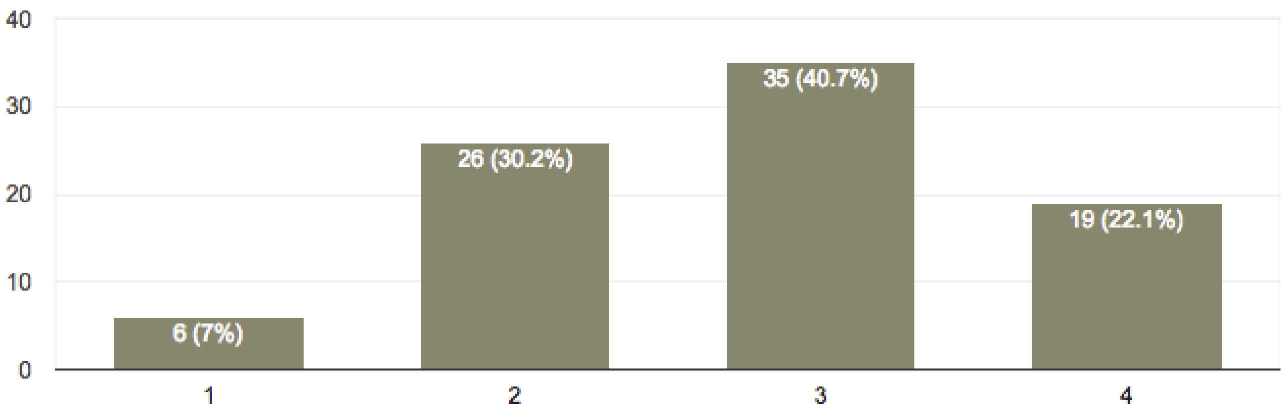
Question 11. I am satisfied with the INTEROPERABILITY of the ASPRS LAS format. (1 is disagree and 4 is agree)

As mentioned, interoperability of LAS implementations has been quite successful. Our question confirms this sentiment, with the majority tending to agree that interoperability of the format is successful.



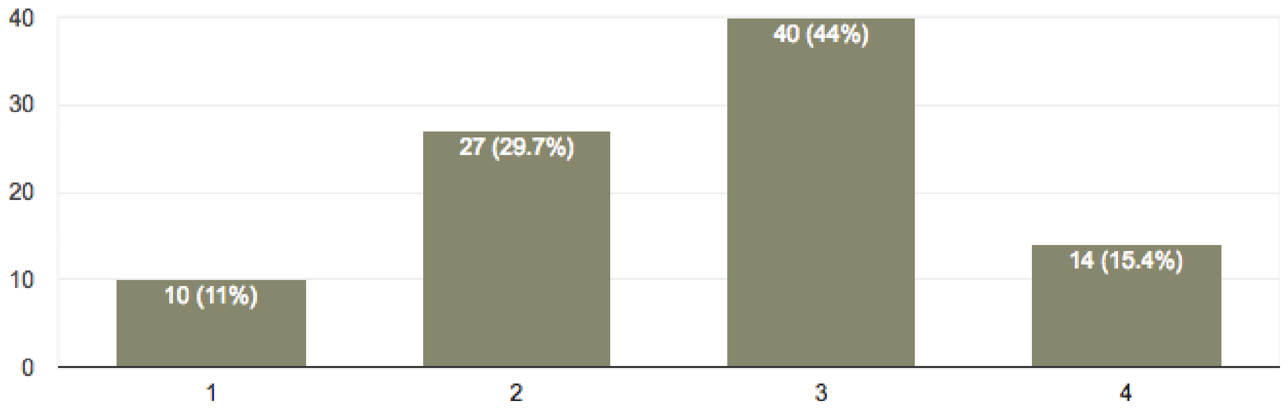
Question 12. I am satisfied with the PERFORMANCE of the ASPRS LAS format. (1 is disagree and 4 is agree)

On the performance topic, the curve starts to slide to the left, as disappointment with compression and the bulk of the format starts to take effect.



Question 13. I am satisfied with the FEATURES of the ASPRS LAS format. (1 is disagree and 4 is agree)

Again, the feature satisfaction of LAS is generally skewed right, but the sentiment is not full satisfaction with the format.

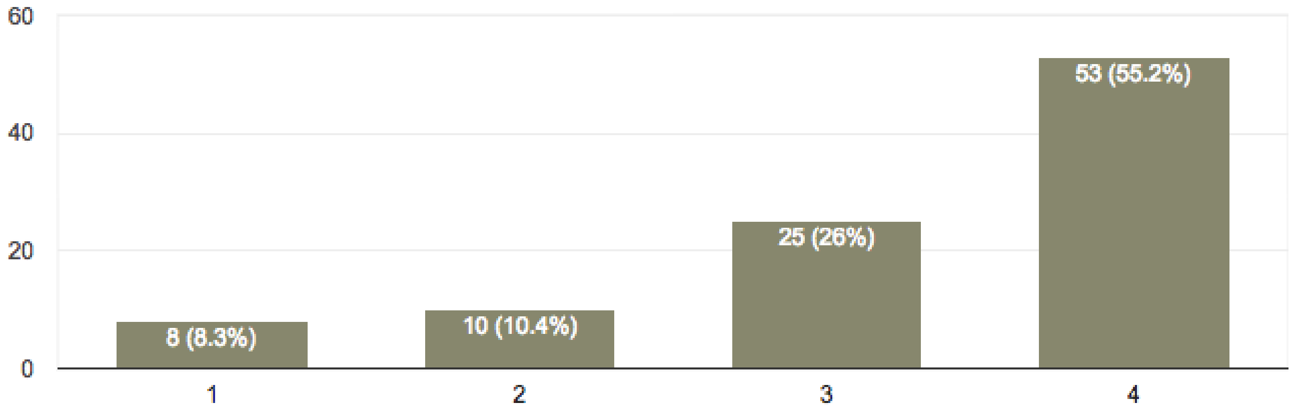


Question 14. Formats commonly in use by the industry are sufficient. (1 is disagree and 4 is agree)

It is likely that the impression of LAS dominates the responses to this question, but sentiment is generally positive in regard to the current formats used by the industry.

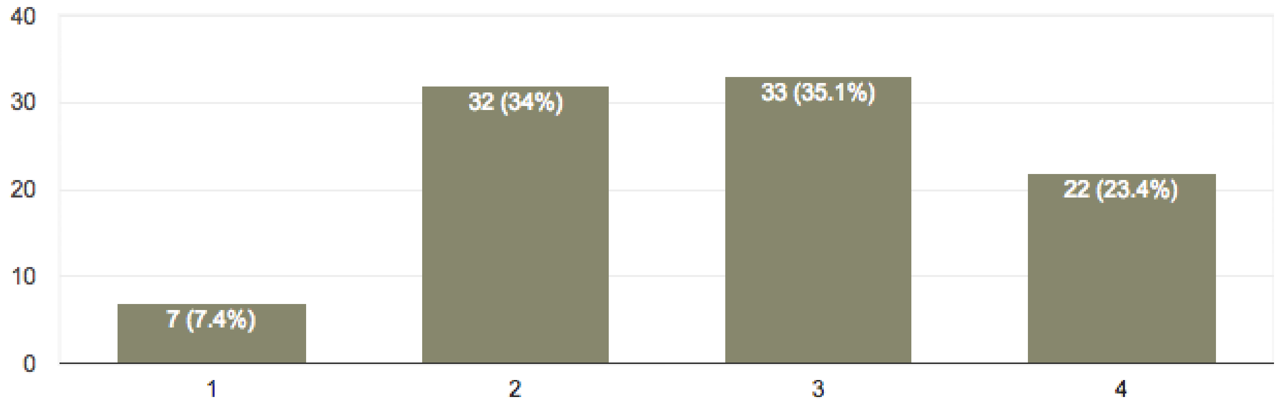
A.3.2. Compression

Compression is an important topic for point cloud users. Not only must the compression be efficient, it must be able to be widely consumed by all software, fast enough to encode and decode, and convenient enough to use for many usage scenarios. The survey highlights that the topic is still on many respondents' minds, and it confirms expectations of where the marketplace currently sits.



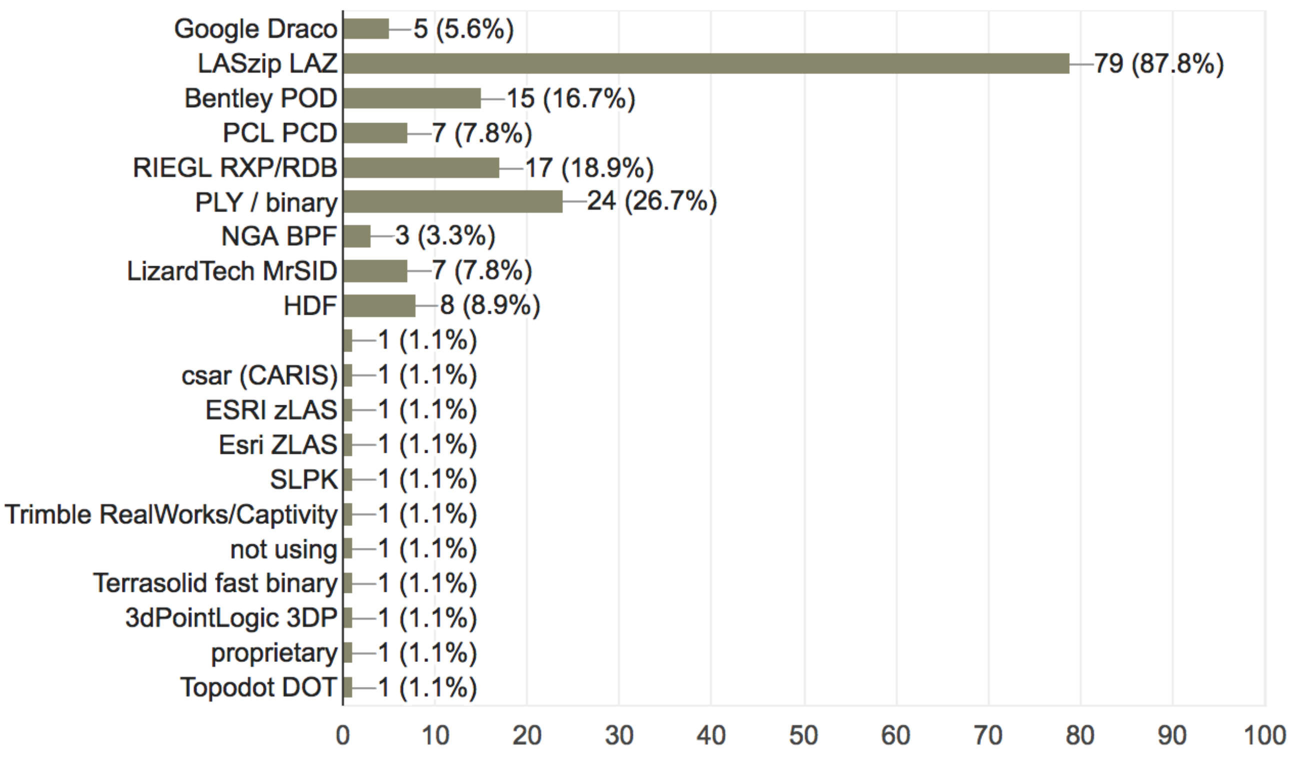
Question 15. Compression is required to effectively utilize point cloud data.

Industry experience indicates that compressed content encodings are important for the consumption and usage of geospatial point clouds. The LAS format represented a "compressed" format of ASCII for many, simply by compacting the binary representation of textual information. LAZ took that further by reducing redundant bits in per-point stored LAS data. The industry reports



Question 16. There are enough point cloud compression options in the marketplace. (1 is disagree and 4 is agree)

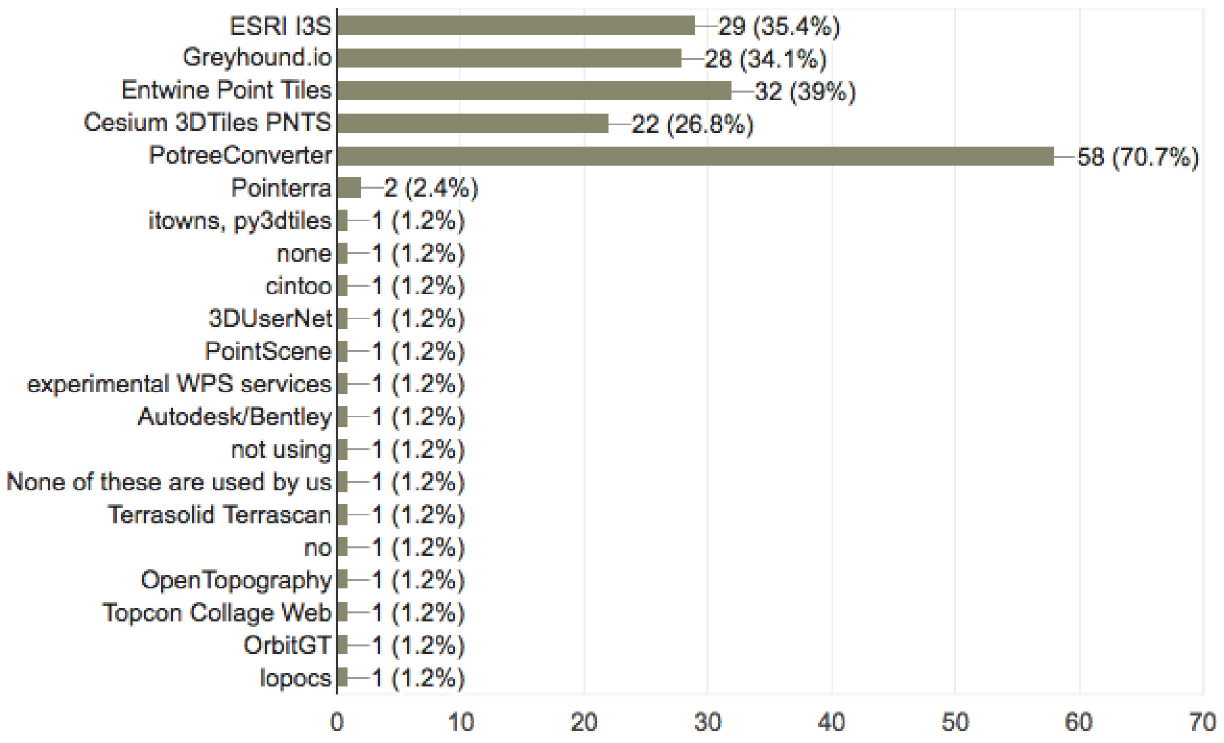
More than half of the respondents agreed that enough point cloud compression options exist, but as with LAS, there is likely to be an undercurrent of unsatisfied audience for the choices that are available. It can be speculated this is due to missing features in the LAS format in relation to spatial partitioning, coordinate system interoperability, and standard compression definitions.



Question 17. Compression software I use

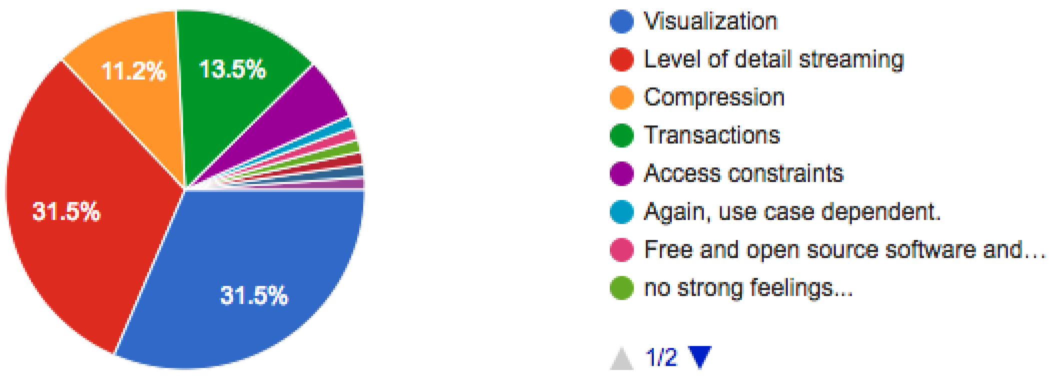
As expected, LASzip dominates the responses, but a wide variety of point cloud compression software use is reported by the respondents.

A.4. Web Services



Question 18. With which point cloud web service software are you familiar? (check all that apply)

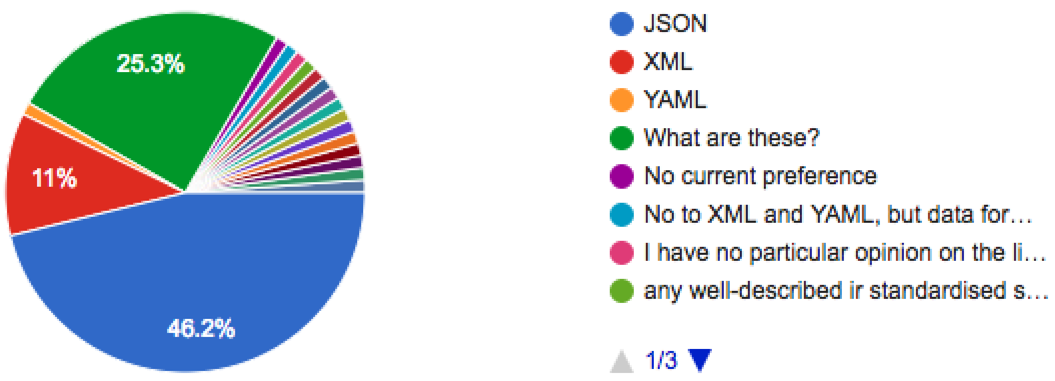
The topic of web services for point cloud data has been frequently identified as a desire, but the market appears to be relatively immature at this time. It is quite likely that the survey did not draw a diverse enough audience on this topic, but Potree and its data conditioning tool PotreeConverter were most well known in the survey.



Question 19. What is the most important topic a point cloud web service should address?

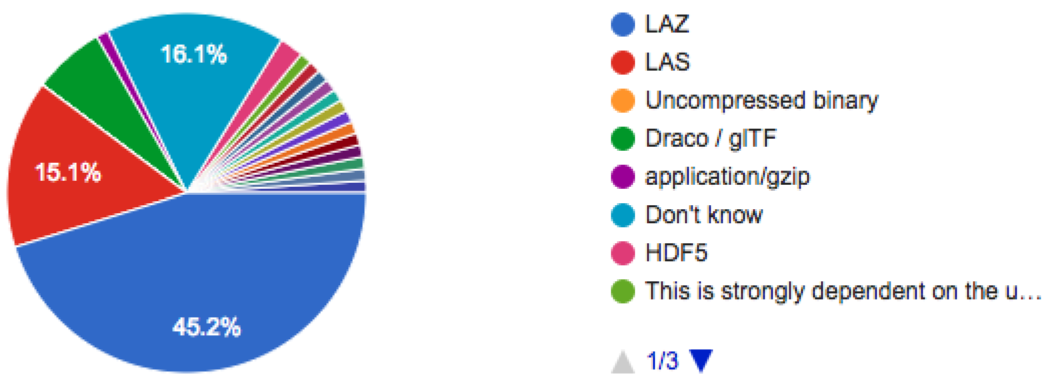
Visualization of point cloud data is an excellent data confirmation tool, but size constraints often make holistic visualization of large collections problematic. Curiously, both visualization and level of detail streaming are identified by respondents as topics that a point cloud web service should address. Presumably, this is due to the challenging nature of keeping data flowing at a visualization environment, and the market's familiarity with visualization capable tools such as Potree.

A.4.1. Transport



Question 20. Which structured data interchange format should point cloud web services use?

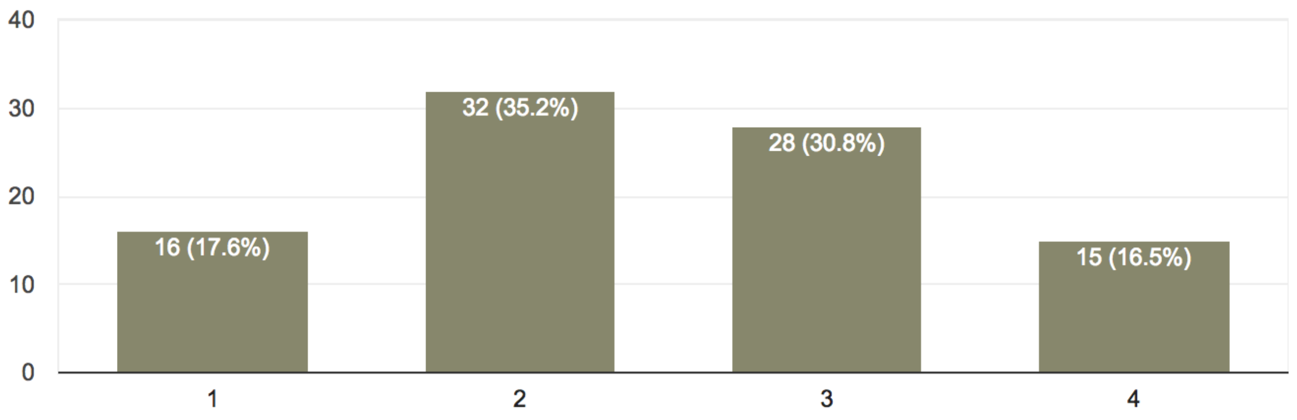
How web services communicate is a topic that can quickly overwhelm. Most importantly the format for protocol chatting is most helpful if it can conveniently interoperate with other protocols. JSON was identified by the survey audience as the most desired on this topic, and this sentiment reflects the current state of web and micro services in use today.



Question 21. Which binary transport format should point cloud web services use?

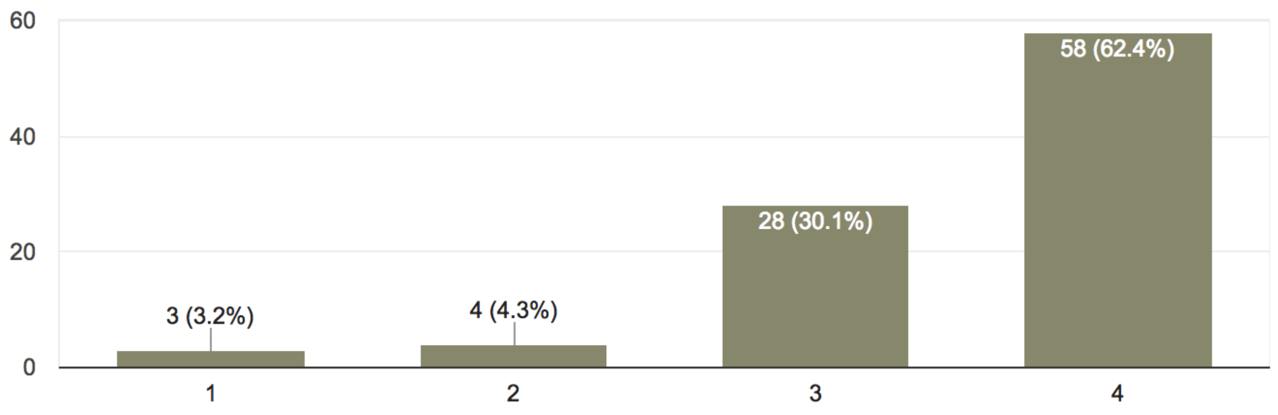
The survey respondents recognized that a text-based data interchange and transport is unlikely to be efficient for a point cloud web service. Although the audience identified LAZ as a primary content encoding, it can be speculated this is due to convenience of available software along with experience with its efficiency.

A.4.2. Protocol



Question 22. HTTP/S is a sufficient transport for point cloud data. (1 is disagree and 4 is agree)

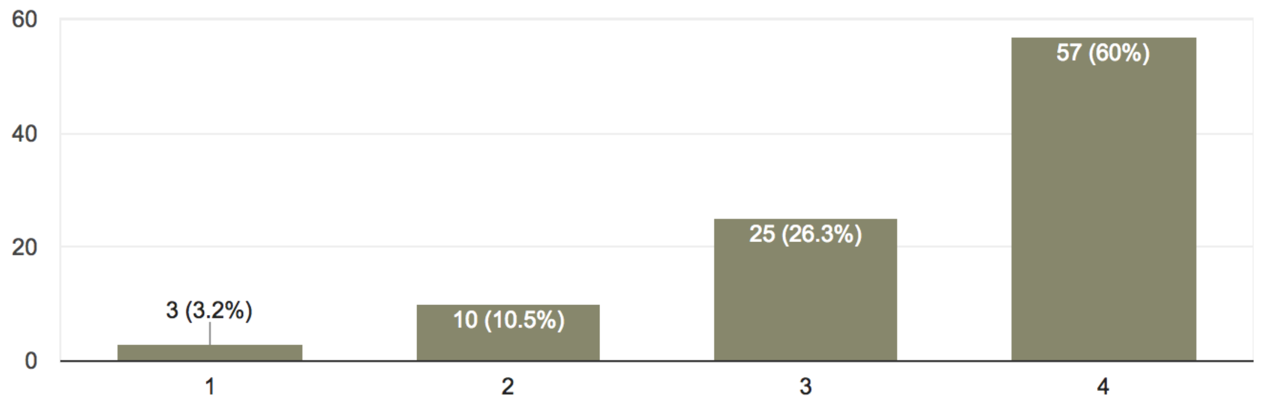
The survey tried to identify whether or not the audience was enthusiastic for a web-based transport, and curiously, the audience is split about in half on the topic. While other streaming approaches are viable (an active socket approach based on WebRTC+WebSockets, for example) the non-transactional nature of point cloud data do tend to fit classical REST-style HTTP interfaces.



Question 23. Spatial partitioning is required for point cloud web services (1 is disagree and 4 is agree)

Spatial partitioning, or the ability to make range queries on the data using a combination of bounding boxes and resolution appears to be a highly desired feature of any point cloud web service approach.

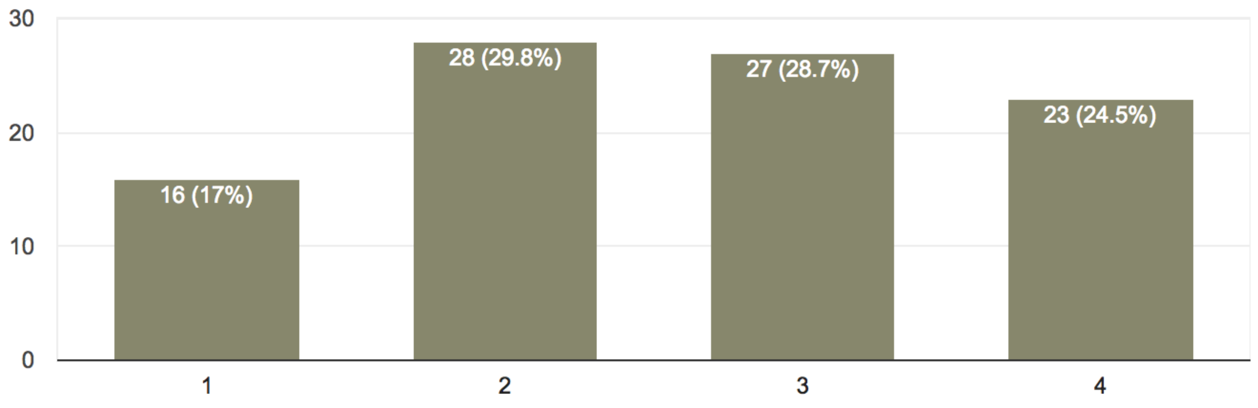
A.4.3. Composition



Question 24. A point cloud web service must support any attribute type. (1 is disagree and 4 is agree)

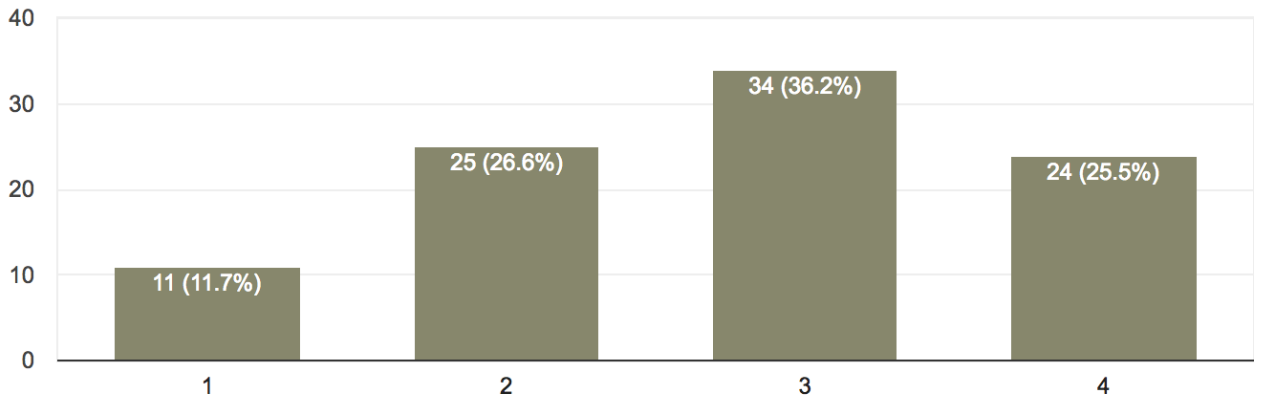
A frequent complaint of ASPRS LAS is that the fixed fields that it requires for attribute storage do not always line up with data captured by scanners and processing toolchains. Respondents appear to desire flexibility with any web service approach in regard to point attributes.

A.4.4. Transactions



Question 25. A point cloud web service must be transactional. (1 is disagree and 4 is agree)

The overhead of transactional web services, and in combination with the volume that point cloud data present, they would be very challenging to implement in a satisfactory way. Respondents seem to agree that the need for point cloud web services to transact data is limited, and this requirement should be relaxed for any point cloud web service implementation standard.



Question 26. Read-only point cloud data access is sufficient for web services. (1 is disagree and 4 is agree)

In combination with transactions, the audience agreed that a point cloud web service does not need to support data update at all, regardless of transactional capability.

Appendix B: Revision History

Table 7. Revision History

Date	Editor	Release	Primary clauses modified	Descriptions
October 1, 2018	Howard Butler	.1	all	draft

Appendix C: Bibliography

1. Wikipedia: Octree — Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/w/index.php?title=Octree&oldid=853700715, (2018).
2. Holzmüller, D.: Efficient Neighbor-Finding on Space-Filling Curves. CoRR. abs/1710.06384, (2017).