Community Standard Justification: 16-133r1

I3S – Indexed 3D Scene Layers

TITLE: I3S - Indexed 3D Scene Layers

CONTRIBUTOR
Name: Keith Ryden

Organization: Esri Inc.
Address: 380 New York Street
Redlands CA

Phone: 909-793-2853 x 1212
Email: kryden@esri.com

DATE: September 20, 2016

Contents

1. Introduction 3
2. Overview of proposed submission 3
   2.1 Designed for 3D 3
      2.1.1 Coordinate Systems 3
      2.1.2 Height Models 3
   2.2 Designed for Web, Mobile and Cloud 4
   2.3 I3S is already an Open Specification 4
   2.4 3D Layer Types 4
   2.5 Data Access and Visualization 6
   2.6 History of the I3S Specification 7
   2.7 Additional References 9
3. Relationship to other OGC and non-OGC standards 10
4. Alignment with OGC Standards Baseline 10
5. Evidence of implementation 10
   5.1 Implementation name: Vricon 10
   5.2 Implementation name: Bentley ContextCapture 12
6. Public availability 13
7. Submitting member(s) 13
8. Intellectual property rights 13
9. Annex A: I3S Technical Overview 14
   9.1 Level of Detail 14
   9.2 Layer Types and Profiles 14
   9.3 Indexing Model 15
   9.4 Geometry Model and Storage 19
   9.5 Textures 19
      9.5.1 Image Formats 19
      9.5.2 Texture Atlas usage and Regions 20
      9.5.3 Texture coordinates 20
   9.6 Attribute Model and Storage 20
   9.7 Level of Detail – LOD Models, Selection Metrics 21
      9.7.1 Discrete LODs 21
      9.7.2 Generation of Discrete Levels of Detail for 3D Objects – A specific Example 22
   9.8 Delivery via REST or as locally consumable Scene Layer Packages 22
   9.9 I3S Flexibility 24
10. Annex B: Additional Examples of Implementation 25

1. Introduction

The Indexed 3D Scene Layers (I3S) delivery format is used to stream 3D geospatial content to mobile, web and desktop clients. This document justifies submission of I3S to the OGC Technical Committee (TC) for consideration to use the OGC Community Standards track. This justification, along with the submitted candidate Community standard, will form the basis for TC review and vote to approve the start of the Community standard process for this standard.

The submitters agree to abide by the OGC TC Policies and Procedures and OGC Intellectual Property Rights Policy (http://www.opengeospatial.org/ogc/policies) during the processing of this submission.

2. Overview of proposed submission

I3S was publicly released in July 2015 as an open specification for streaming large, heterogeneous geospatial data sets with 3D content. 3D content can include discrete 3D objects, large continuous meshes, 3D vector points, point clouds, and other content across enterprise systems that may consist of server components, cloud hosted components, and a variety of client software from desktop to web and mobile applications. I3S supports storage and transmission of very large data sets that may consist of millions of discrete 3D objects with attributes as well as integrated surface meshes that may cover many thousands of square kilometers of the Earth's surface.

The I3S specification can be viewed at <https://github.com/esri/i3s-spec>. The next section describes key design and operational aspects of the specification. A detailed overview of the I3S conceptual model is provided in Annex A.

2.1 Designed for 3D

I3S is specifically designed to support 3D geospatial content and supports the requisite coordinate systems and height models in conjunction with a rich set of layer types.

2.1.1 Coordinate Systems

I3S supports both 3D Global Coordinate Systems as well as locally applicable 3D Map Coordinate systems that are based on well-known map projections for the x-y plane along with additional height information for the z axis.

2.1.2 Height Models

The specification accommodates declaration of a vertical coordinate system that may be ellipsoidal (elevation/ height defined with respect to a reference ellipsoid) or orthometric (elevation / height defined with respect to a reference geoid / gravity surface). This allows I3S to be applied across a diverse range of fields and applications where the particular definition of elevation/height is of importance.

2.2 Designed for Web, Mobile and Cloud

I3S is designed from the ground up to be cloud, web and mobile friendly. I3S is based on JSON, REST and modern web standards and is easy to handle, efficiently parse and render by Web and Mobile Clients. I3S is designed to stream large 3d datasets and is designed for performance and scalability.

2.3 I3S is already an Open Specification

I3S was introduced as an open specification for the purpose of encouraging community adoption, encouraging feedback, and for ensuring that organizations implementing the specification would have flexibility in access to 3D data. The specification is currently licensed under the *Creative Commons Attribution-NoDerivs 3.0 Unported License*. Implementers can use the specification in services, clients or processing tools without restrictions.

2.4 3D Layer Types

A single I3S data set, referred to as a *Scene Layer* is a container for arbitrarily large amounts of heterogeneously distributed 3D geospatial data. An I3S Layer is characterized by a combination of *layer type* and *profile* that fully describe the behavior of the layer and the manner in which it is realized within the specification.

The format is declarative and extendable and has been used to represent different types of 3D data.

The following layer types have been specified and validated via implementation and production deployments:

- **3D Objects**
  - example : Building Exteriors
  - sources : Derived from GIS Data, as well as 3D models in various formats

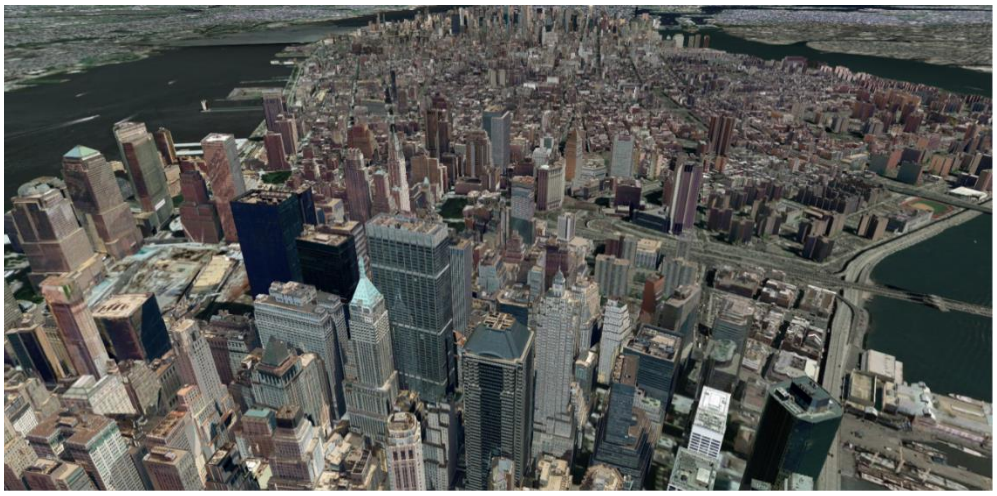


Figure 1: View of Manhattan textured buildings – exhibiting a wide range of distance and LODs

- **Integrated Meshes**
  - Example: Mesh surface representing the skin of the Earth, including vegetation, buildings and roads
  - Sources: Derived from satellite, aerial or drone imagery via dense matching photogrammetry, or calculated

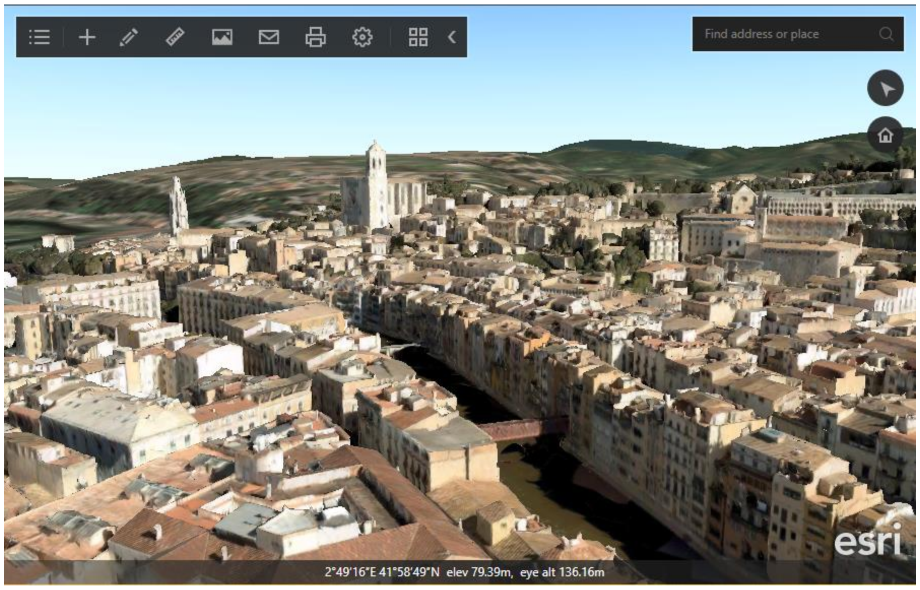


Figure 2: Integrated Mesh of Girona, Spain

- **Points**
  - Examples: Hospitals, schools, trees, cars
  - Sources: feature locations combined with Instanced 3D models generated by hand

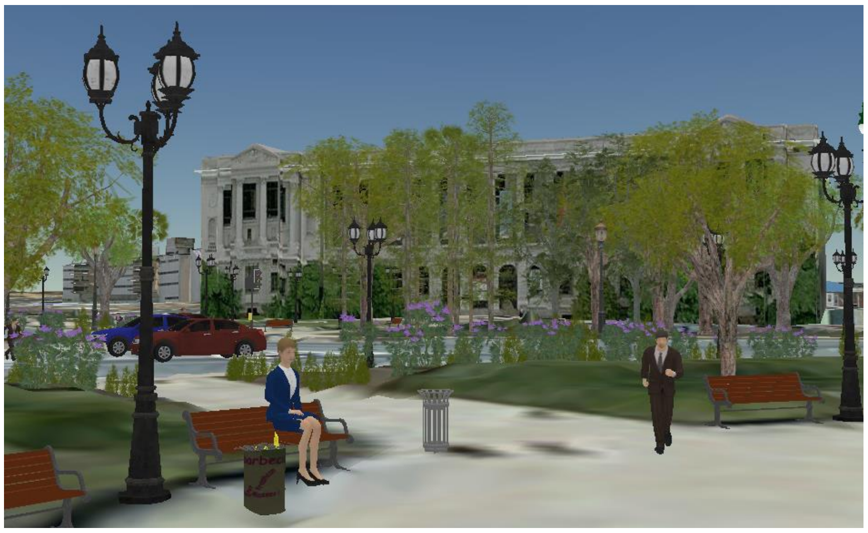


Figure 3: Street furniture, vegetation, people, and vehicle represented as symbolized points

The following layer types are in development process with release of the specification in Q4 2016 and release of a production implementation in Q4 2016.

- **Point Clouds**
    - Example: LiDAR data sets
    - Sources: Typically sensor-collected or Photogrammetrically derived

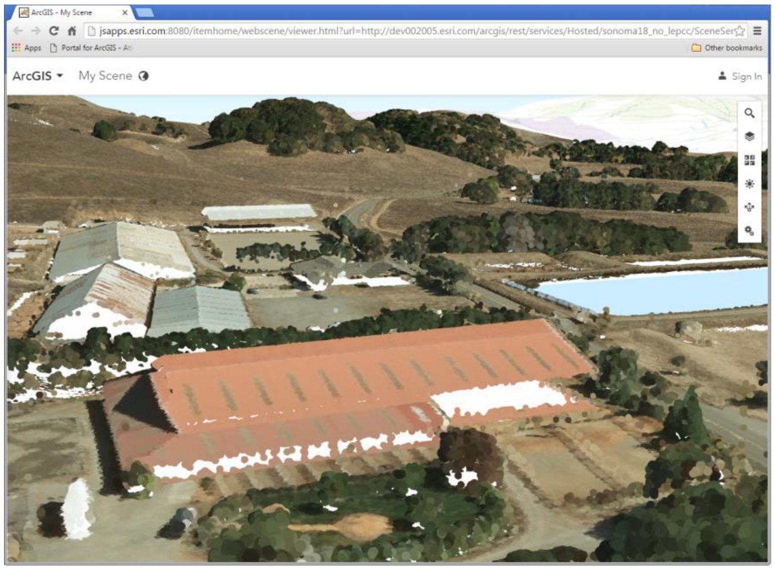


Figure 4: LiDAR-collected Point Cloud displayed in a web browser as a scene layer

The following layer types are planned for future inclusion:

- **Lines** (e.g. Roads and Pipes; from GIS Data)
- **Polygons** (e.g. Soil or Forest Cover; from GIS Data)

2.5 Data Access and Visualization

I3S Scene Layers are designed to provide clients access to data, regulated by visual extent and camera distance and type-dependent indexing. Clients have the ability to adjust or override visualization properties of layers independently according to their needs and according to the nature of the layer's data. Data here refers to both the geometry and the attributes for 3D Object, Point, Line and Polygon Features as well as the vertex geometry and vertex attributes for vertices within layers that represent geographic fields, like with integrated meshes and point clouds.

The degree of separation between data and implied visualization varies depending on the type of layer – at one extreme an integrated mesh layer, representing [in effect] "3d imagery", has a strong implication that clients will display the mesh geometry using the textures that are advertised by and available with the layer. At the other end layers representing 3D features/entities including space occupying 3D objects as well as Points carry a full representation of their attributes allowing clients to thematically map and visualize the data according to their needs. In both cases the I3S access API separates access to geometry from access to textures and attributes, giving clients needed flexibility. In many cases the geospatial features' data that is part of the I3S layer will itself be a cached, cooked or preprocessed version (the industry uses a variety of terms to denote the process of generating the 3D access optimized representation) of primary data that resides in a transactional system of record.

2.6 History of the I3S Specification

I3S was released on Github as an open specification on April 30, 2015 [1].

The initial public specification included a general description of i3S along with a description of the profile for storing and streaming 3D Object scene layers, which may be any kind of monolithic or discrete 3D model that may have a consistent set of attribute information across the entire data set.

Additional profiles for 3D vector-based points and integrated meshes were introduced in software implementation in multiple releases leading up to June 2016 and were updated in the open specification September 12, 2016.

Future spec releases are planned to incorporate feedback from implementers as well as validation tools to streamline the implementation workflow.

The Integrated Mesh, 3D Object, and 3D Point layer types are in production use as part of deployed ArcGIS systems, including browser, mobile and desktop clients working against on-premises and cloud deployed servers.

The Point Cloud layer type has been developed and is currently undergoing beta testing. It is planned for incremental release across the range of supported clients starting in Dec 2016.

Esri offerings

Esri started demonstrating I3S capability in 2013. On July 16, 2015, ArcGIS Pro 1.1 was released which could publish 3D Object scene layers to ArcGIS for Server 10.3. Users could also follow a packaging and upload workflow to transfer scene layers from ArcGIS for Server to ArcGIS Online.

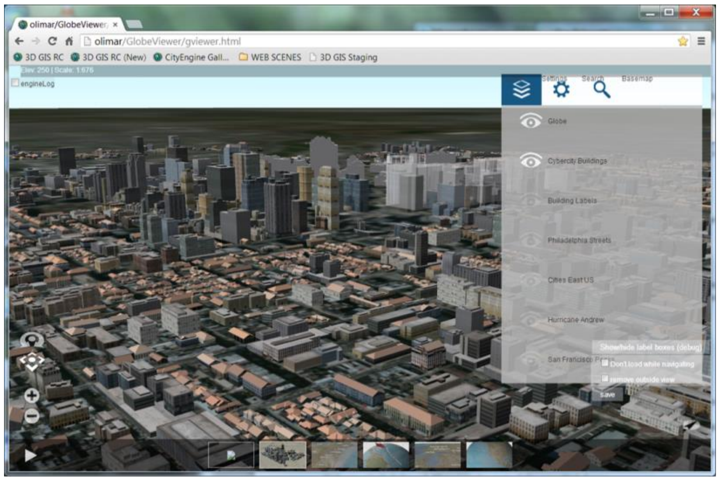


Figure 5. Demonstration scene layers shown on stage at the 2013 Esri International User Conference

With ArcGIS for Server 10.4 and ArcGIS Pro 1.2 in February and March 2016, users could also share 3D Point scene layers directly from ArcGIS Pro to ArcGIS for Server. The ArcGIS Online JavaScript-based web scene viewer was able to demonstrate the capability to display demonstration scene layers shared by Esri starting in December 2014.

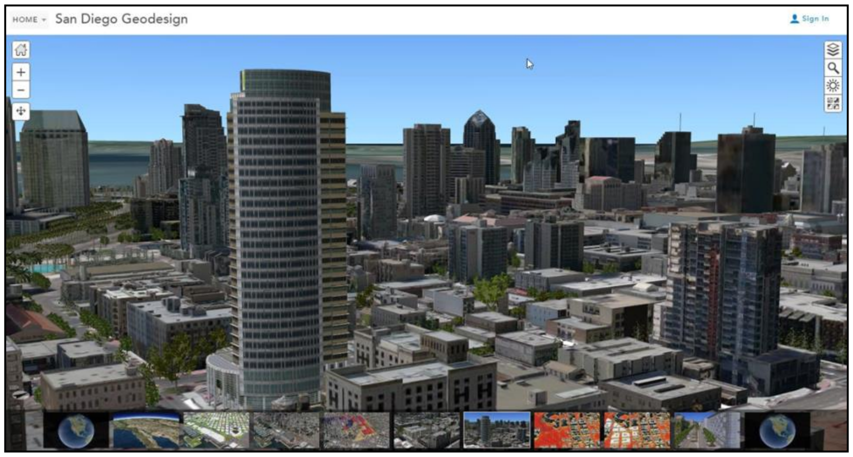


Figure 6. I3S content shown in the officially supported 3D scene viewer served by ArcGIS Online shown from early 2015

With the release of ArcGIS Earth 1.0 in January 2016, non-GIS users could easily access 3D object scene layers shared through ArcGIS Online or ArcGIS for Server. Esri's JavaScript API supported custom web interfaces that could access scene layers in beta in July 2015 and in production in May 2016. Web AppBuilder 2.0 entered beta with the ability to consume and display scene layers in configured web apps in April 2016 and entered production in July 2016.

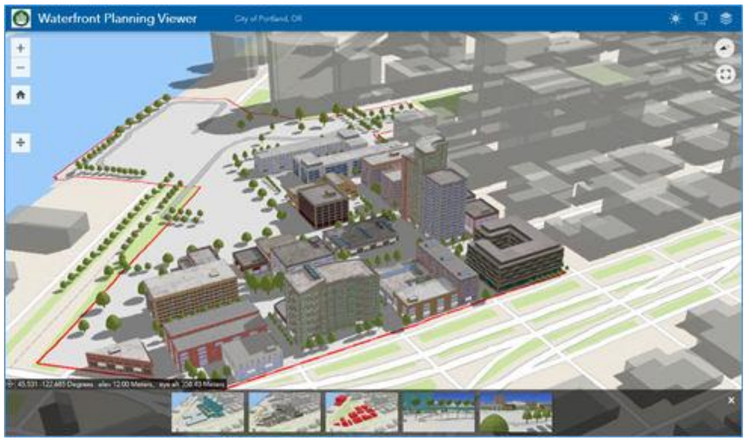


Figure 7. Scene layers used in a Web AppBuilder project from May 2016

In early 2016, data providers began engaging with Esri to start providing production-quality content in I3S format. Most notably these providers include Vricon and Bentley Systems.

Vricon

In May 2016, Vricon was able to successfully begin creating and testing I3S integrated mesh scene layer content. Vricon scene layers are typically created as multi-LOD scene layer packages covering approximately 10,000 km in area and typically ranging from 40 up to 120 GB in size when packaged on disk. More information is provided below.

Bentley Systems, Incorporated

Also in May 2016, Bentley Systems released ContextCapture with the ability to create integrated mesh scene layers from and uploaded demonstration services to ArcGIS Online. More information is provided below.

2.7 Additional References

[1] https://github.com/esri/i3s-spec

[2] https://s3.amazonaws.com/webapps.esri.com/esri-proceedings/devsummit14/papers/dev-013.pdf

http://proceedings.esri.com/library/userconf/devsummit15/papers/dev_int_189.pdf

[3] http://proceedings.esri.com/library/userconf/devsummit16/papers/dev_int_01.pdf

[4] SIGGRAPH 2016 Cartography Birds of a Feather

[5] https://www.linkedin.com/pulse/bentleys-contextcapture-now-supporting-esri-i3s-3d-scene-john-taylor

3. Relationship to other OGC and non-OGC standards

Implementation of I3S is not dependent on implementation of other OGC standards. That said, the 3D Portrayal SWG has a work item to define and document a Best Practice for using I3S with the 3DPS interface.

Optional elements, such as textures using the S3 Compression Format (DXT), may be included as payload content for better performance on specific devices or platforms.

Currently, no OGC standards reference i3S.

I3S shares the *Array Buffers* and *Array Buffer Views* Khronos Typed Array specification with glTF.

4. Alignment with OGC Standards Baseline

I3S complements the OGC ecosystem in the following ways:

- I3S provides an open, multi-platform standard for transmission of data within the framework of the OGC 3DPS
- I3S can be used alongside OGC WMS, WMTS, KML and other OGC formats to create 2D and 3D mapping visualizations and experiences
- I3S serves as a common tool to package and disseminate, as files or as services, a variety of GIS content including large integrated meshes, entity/feature-centric 3D models, vector GIS features, and point clouds
- I3S may be extended to accommodate additional data types being developed and created by the GIS community
- I3S is already broadly accessible to the Esri-related segment of the GIS community through production software implementations and supports interoperability between that community segment and others

5. Evidence of implementation

The following implementations use the proposed Community standard. Two examples are provided in the main body of this document. More examples are provided in Annex B of this document.

5.1 Implementation name: Vricon

Date of most recent version: Currently able to provide data as i3S packages

Implementation description: Vricon is creating the highest resolution textured mesh of the globe from satellite data. Their process includes creating geocells of data that are approximately 10,000 km sq. These geocells may be created as single scene layer packages and typically range from about 60GB in size up to about 120GB.

Vricon geocell scene layers packages may be uploaded and shared through ArcGIS Online or ArcGIS Server or they may be viewed locally in ArcGIS Earth.

Implementation URL:
http://www.arcgis.com/home/webscene/viewer.html?layers=1f97ba887fd4436c8b17a14d83584611

Is implementation complete? ☐ **Yes** [☒ **No**]

If not, what portions of the proposed Community standard are implemented?

Vricon has currently implemented only the Integrated Mesh scene layer profile.

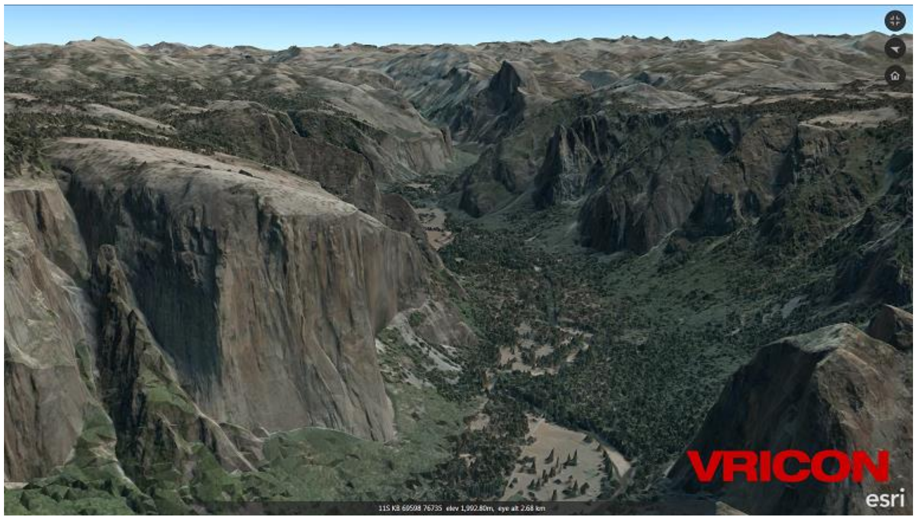


Figure 8: Vricon integrated mesh scene layer shown in ArcGIS Earth.

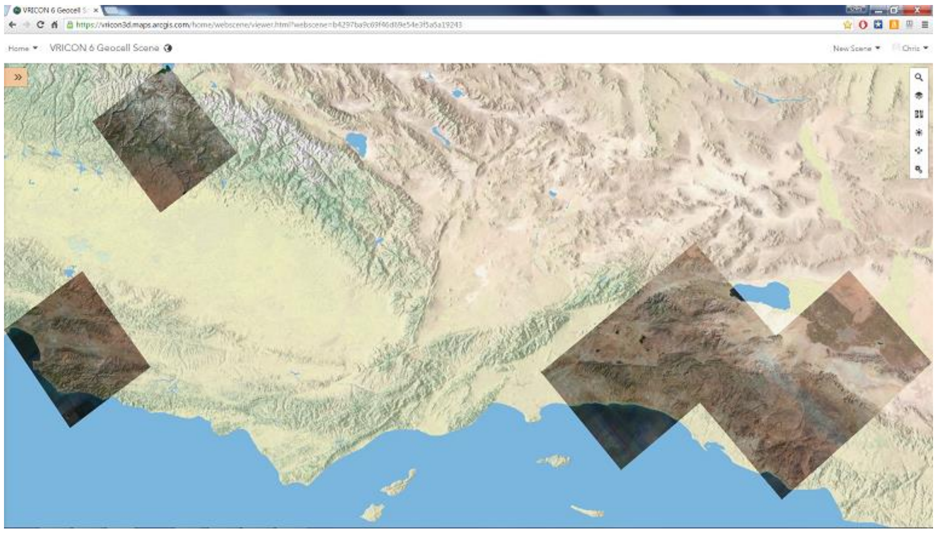


Figure 9. Web scene showing six Vricon geocells covering approximately 60,000 sq km and totaling over 400GB of disk space in packaged format

5.2 Implementation name: Bentley ContextCapture

Date of most recent version: 2016

Implementation description:

With ContextCapture, you can quickly produce even the most challenging 3D models of existing conditions for infrastructure projects of all types, derived from simple photographs. Without the need for expensive, specialized equipment, you can quickly create and use these highly detailed 3D reality models to provide precise real-world context for design, construction, and operations decisions for use throughout the lifecycle of projects.

ContextCapture currently is able to export scene layer packages of integrated mesh data that can be uploaded and shared through ArcGIS Online and ArcGIS for Server.

Implementation URL:
http://www.arcgis.com/home/webscene/viewer.html?layers=423ead7f6deb48e9a74535e130e0624e

Is implementation complete? Yes [✓ No]

If not, what portions of the proposed Community standard are implemented?

Bentley has implemented the Integrated Mesh profile only. Also, the current release has an error that prevents Scene Layer Packages from being used as local files unless they are repackaged without compression.

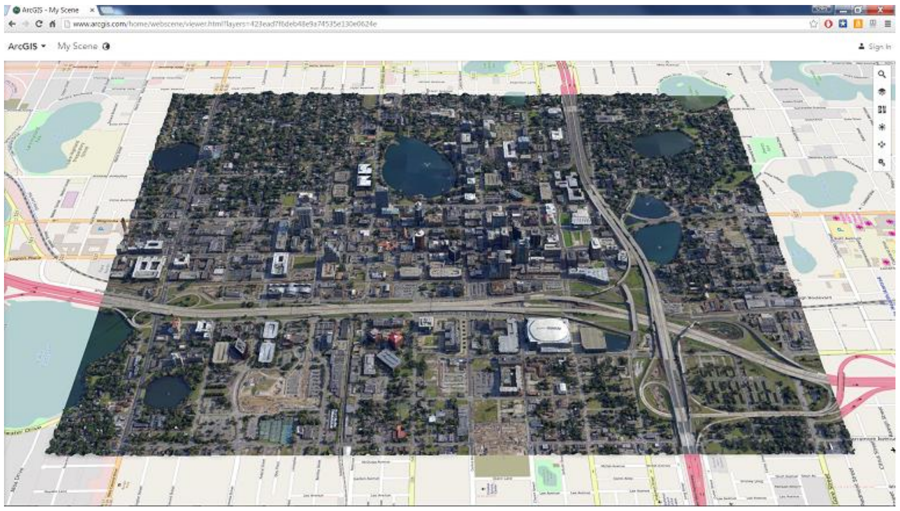


Figure 2. Integrated mesh scene layer from Bentley Systems' ContextCapture showing in a web scene.

See **Annex B** for more evidence of implementation.

6. Public availability

Is the proposed Community standard currently publicly available? **Yes**
URL: https://github.com/Esri/i3s-spec

7. Supporting member(s)

- Esri
- Allan W. Shearer, M.L.A., Ph.D., The University of Texas at Austin
- Volker Coors, Hochschule für Technik Stuttgart
- 52North
- Carl Reed, Carl Reed and Associates
- Esri Canada
- Esri India Technologies Ltd.
- Vricon, Inc.
- interactive instruments GmbH
- AAM Pty Ltd
- Wetransform GmbH
- CAE

Esri invites other interested OGC members to contact Keith Ryden (kryden@esri.com) to be added to the list of supporting members.

8. Intellectual property rights

Will the submitter retain intellectual property rights? [✓ **Yes**] ☐ **No**

If yes, the submitter will be required to work with OGC staff to properly attribute the submitter's intellectual property rights.

If no, the submitter will assign intellectual property rights to the OGC.

9. Annex A: I3S Technical Overview

9.1 Level of Detail

The concept of Level of Detail (LOD) is intrinsic to the specification. Scene Layers may include levels of detail that apply to the layer as whole and serve to generalize or summarize information for the layer, similar to image pyramids and also similar to raster and vector tiling schemes. A node in the I3S scene layer tree could be considered the analog of a tile in a raster or vector tiling scheme. Scene layers support levels of detail in a manner that preserves the identity and representation of the individual features that are retained within any level of detail.

9.2 Layer Types and Profiles

Layers are described using two properties, type and profile. The type of a layer describes its semantic meaning to the consumer as described in the section above. The profile for a layer includes additional detail on the specific I3S implementation for the layer that is exposed to clients. In most cases the relationship between layer and profile is 1:1 but in certain cases multiple layers that represent semantically different types of information can make use of the same underlying profile. In other cases the same layer type can support multiple profiles optimized for different use cases. The following table shows the layer types and associated profiles, indicates which layers are used to represent features (entities) vs geographic fields, and also includes information on support for feature or vertex attributes for the layers.

Layer Type *(example)*	**Profile**	**Features with Identity**	**Attributes**
3D Object	*mesh-pyramids*	Yes	Yes
Integrated Mesh	*mesh-pyramids*	No	*planned :* Triangle Attributes
Point	*points*	Yes	Yes
Pointcloud	*pointclouds*	No	Vertex Attributes
Line	*lines*	Yes	Yes
Polygon	*polygons*	Yes	Yes

Organization and structure

I3S organizes information using a hierarchical, node-based spatial index structure in which each node's payload may contain features with associated geometry, textures and attributes.

9.3 Indexing Model

I3S is agnostic with respect to the model used to index objects / features in 3D space. Both regular partitions of space (e.g. quadtrees and octtrees) as well as density dependent partitioning of space (e.g. R-Trees) are supported. The specific partitioning scheme is hidden from clients who navigate the nodes in the tree via REST API. The partitioning results in a hierarchical subdivision of 3D space into regions represented by nodes, organized in a bounding volume tree hierarchy (BVH). Each node has an address based on its tree-key.

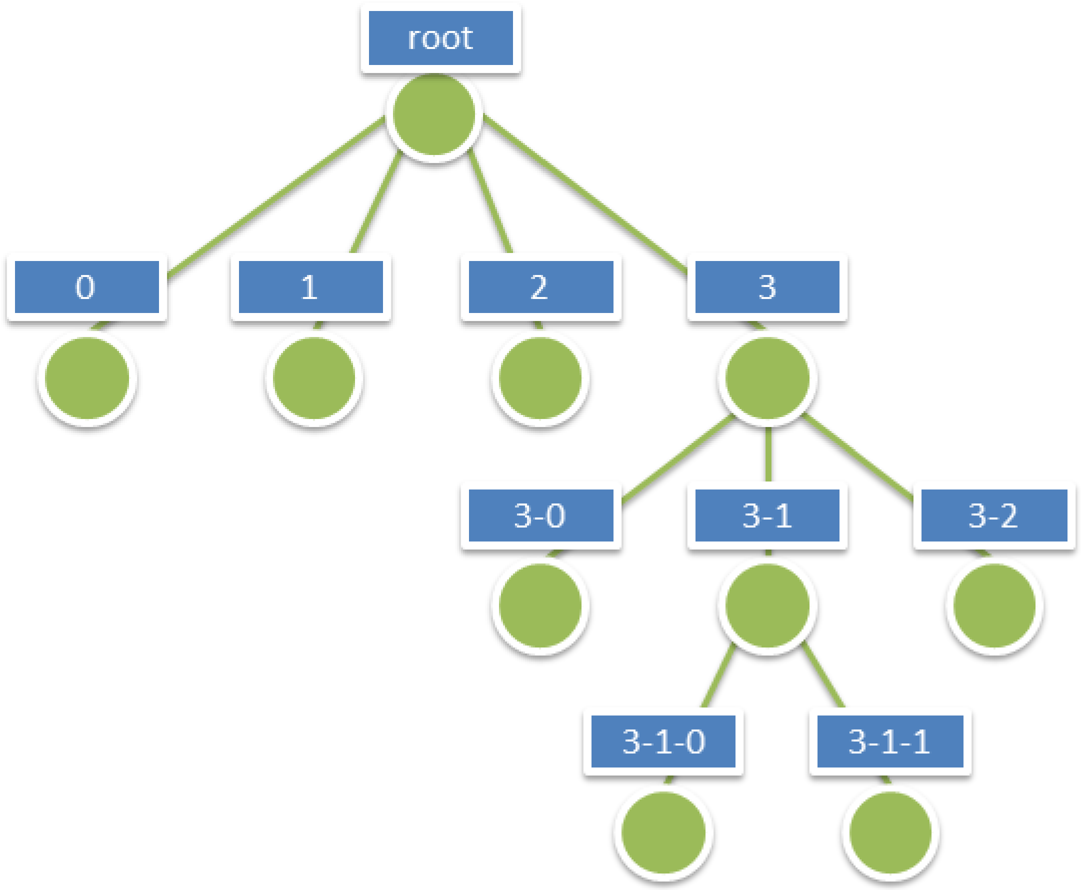


Figure 11: I3S node hierarchy

The information for a node is stored in multiple individually accessible resources. The node index document is a lightweight resource that captures the BVH tree topology for the node, in addition to the node's bounding volume and meta-data used for LOD switching metrics. This resource allows for tree traversal without the need to access the more voluminous content associated with a node (geometry, texture data, attributes), where the decision to render the node is based on node's bounding-volume visibility in the current 3D view and a visual quality determination made by the client using the information included in the node index document. The node's quality is estimated as a function of current view parameters, the node's bounding volume, and LOD Selection threshold value of the node.

The specification supports both bounding spheres (MBS) and oriented bounding boxes (OBB) as a node's bounding volume.

Each interior node logically contains or covers the set of information covered by the nodes below it and participates in a path to the leaf nodes below it. Interior nodes may contain generalized or reduced detail versions of information contained in descendent nodes.

The physical organization of information within nodes is as follows:

- The node index document is a lightweight resource that describes the topology of the node within the tree and includes references to other sub-resources.
- The feature-data sub-resource for a node is a text resource that contains the identifiers for the set of features within a node. It stores the geometry and attributes for all of the features in the node by value or as references to sub-resources.
- The geometry, attribute and texture sub-resources describe the geometry, attribute and texture for the node. Geometry and attribute sub-resources represent the geometries and attributes of all of the features within the node and include the identifiers of the owning features within the node as well as the mapping between individual feature identifiers and their geometry segments. Vertices within the geometry contain the appropriate texture coordinates.

An I3S profile can have either a single text-based feature-data sub-resource that contains all geometry and attribute information, or separate, binary and self-contained geometry and attribute sub-resources. Applications accessing the latter do not need to first fetch the feature-data resource in order to interpret them.

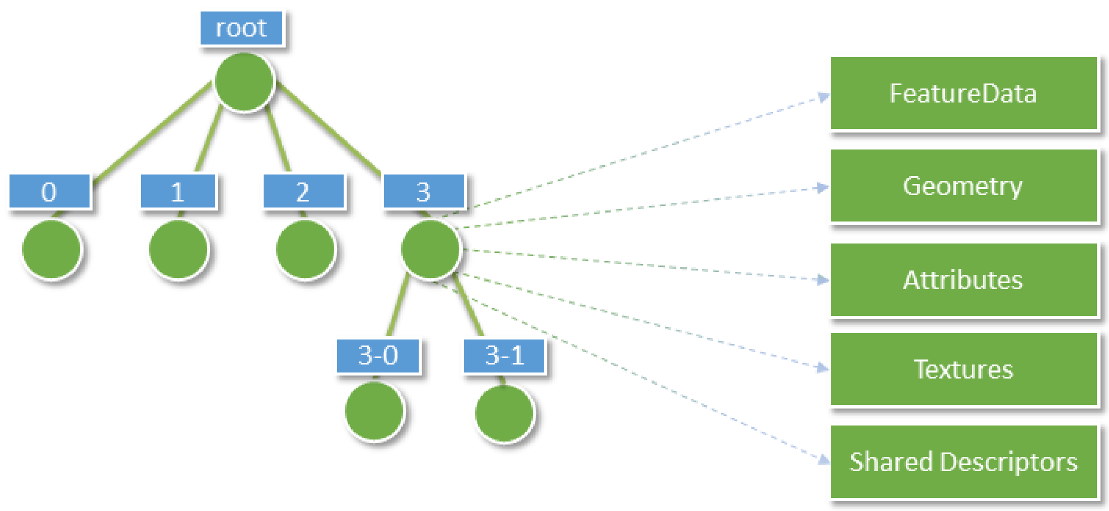
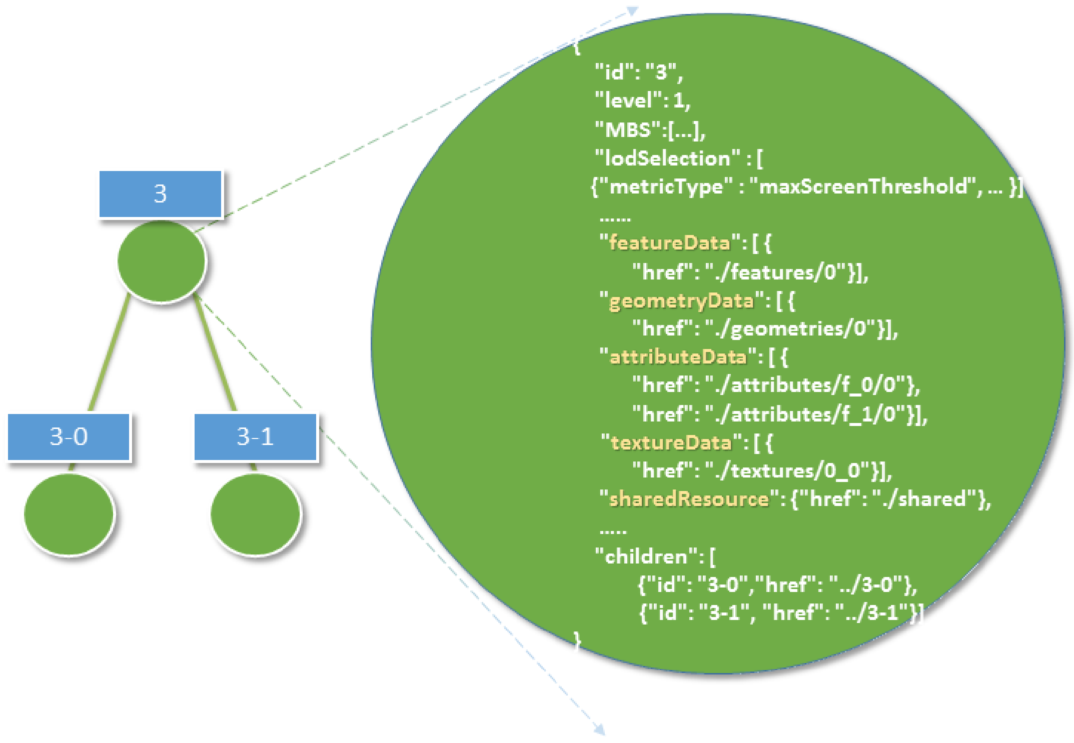


Figure 12: Per-node I3S resources: Feature-Data, Geometry, Attributes, Textures, Shared-Descriptors



```
{
  "id": "3",
  "level": 1,
  "MBS":[...],
  "lodSelection" : [
 {"metricType" : "maxScreenThreshold", ... }]
  ......
  "featureData": [ {
      "href": "./features/0"}],
  "geometryData": [ {
      "href": "./geometries/0"}],
  "attributeData": [ {
      "href": "./attributes/f_0/0"},
      "href": "./attributes/f_1/0"}],
  "textureData": [ {
      "href": "./textures/0_0"}],
  "sharedResource": {"href": "./shared"},
  .....
  "children": [
       {"id": "3-0","href": "../3-0"},
       {"id": "3-1", "href": "../3-1"}]
}
```

Figure 12a: I3S node content

Figure 6 below shows the node tree of an Indexed Scene Layer whose layer type is *3D Objects* and whose profile is *mesh-pyramid*. Nodes are in blue, the numbers within the blue boxes represent the identifier or address for each node. The orange boxes indicate the features explicitly represented within the node, the numbers represent feature identifiers. Each node has associated geometry, texture and attribute resources that compactly store the geometries, attributes and textures of all of the features explicitly represented by the node, as typed arrays and texture atlases. The turquoise boxes show the geometry resource associated with each node, attribute and texture resources have been deliberately omitted. Each geometry resource is an array of geometries. The geometry resource also stores the mapping from each feature to the set of geometry elements within the larger geometry resource which is stored in a compact manner similar to run length encoding. A similar storage model is used for the attributes and textures associated with the features explicitly stored within a node.

Each node contains explicit references (the green lines) to the child nodes below it in the bounding volume hierarchy. Each node logically covers all of the features covered by the nodes in its sub-tree, though only some of them may be explicitly represented within the node. Applications make the decision on using the representation within the node vs descending to more detailed nodes. The specific example illustrated shows the situation within node “3” where “Feature 6” has been generalized away at this lower level of detail and is intentionally no longer explicitly represented within the payload of node 3.

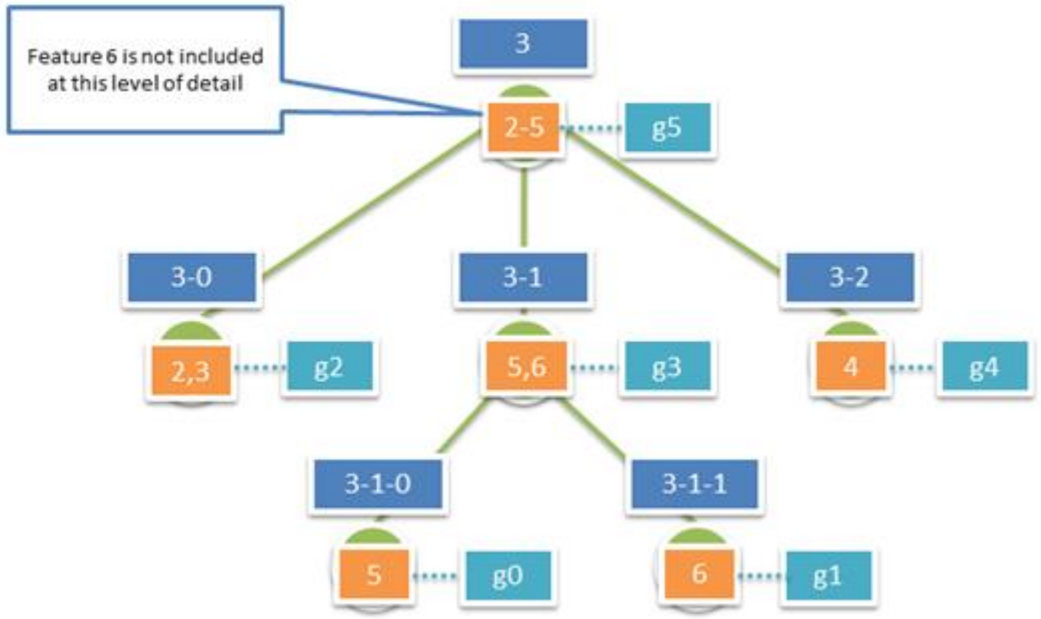


Figure 13: Example Nodes in a Mesh Pyramid. Orange boxes represent features stored explicitly within the node, the numbers represent feature identifiers. Turquoise boxes represent the geometry instances associated with each node – each geometry instance is an aggregate geometry (a geometry collection) that covers all the features in the node.

The node index document and associated metadata use JSON as the representation for storage, while more voluminous data, like Geometry or Feature Attributes, use binary representations in the form of *Array Buffers* and *Array Buffer Views* (following the Khronos Typed Array specification). These authoring decisions are captured as metadata for the layer as a whole.

9.4 Geometry Model and Storage

All Scene Layer types make use of the same fundamental set of geometry types:

- points
- lines
- triangles

Geometries use binary storage and consumption representation, controlled by *Array Buffer View* geometry property declarations. I3s provides full control over those properties, such as per-vertex layout of components (e.g. position, normal and texture coordinates), in order to ensure the same pattern for face and vertex elements across the Scene Layer.

I3S supports storage of triangle meshes via the *triangles* geometry type.

Both *3D Objects* and *Integrated Mesh* layer types model geometries as triangle meshes using the *mesh-pyramids* profile. The mesh-pyramids profile uses the *triangles* geometry type to store triangle meshes with reduced level of detail representations of the mesh, segmented by features, available in the interior nodes as described above.

In the future, polygons would be another layer type that would use triangle meshes and the *triangles* geometry type. Polygonal feature geometries would be tessellated and stored as triangle meshes within I3S, retaining the feature to geometry mapping so that clients could render features and reconstitute polygonal geometries as needed.

9.5 Textures

Textures are stored as a binary resource associated with a node. The texture resource for a node contains the images that are used as textures for the features stored in the node. The *mesh-pyramids* profile supports either a single texture or a texture atlas per node.

9.5.1 Image Formats

The following default texture formats are recommended by the I3S specification: JPEG for RGB and PNG for RGBA. These defaults were chosen because of low bandwidth consumption and widespread adoption in all steps of the tool chain, such as supporting alpha transparency rendering in WebGL.

I3S allows authoring applications to provide additional texture formats via '*textureEncoding*' declarations that use MIME types. For example, most existing I3S services provide "image/vnd-ms.dds" (with S3TC inside) in addition to the default "image/jpeg" encoding.

9.5.2 Texture Atlas usage and Regions

Individual textures, when aggregated into texture atlases, become sub-textures described by sub-image '*Regions*'.

The pixels belonging to a sub-texture are identified by the *subimageRegion*: [umin, vmin, umax, vmax] vertex attribute. Region information is meant to be passed on to the shader using a separate vertex attribute so that every vertex UV coordinate becomes an UVR coordinate, with the R encoding the [umin, vmin, umax, vmax] of the region.

9.5.3 Texture coordinates

Texture coordinates do not take atlas regions into account directly. They always range from 0...1 in U and V, except when using the "repeat" wrapping mode, where they may range from 0...n (n being the number of repeats). The client is expected to use the *subimageRegion* values and the texture coordinates to best handle repeating textures in atlases. This approach has been selected since client capabilities in dealing with more complex UV cases vary greatly.

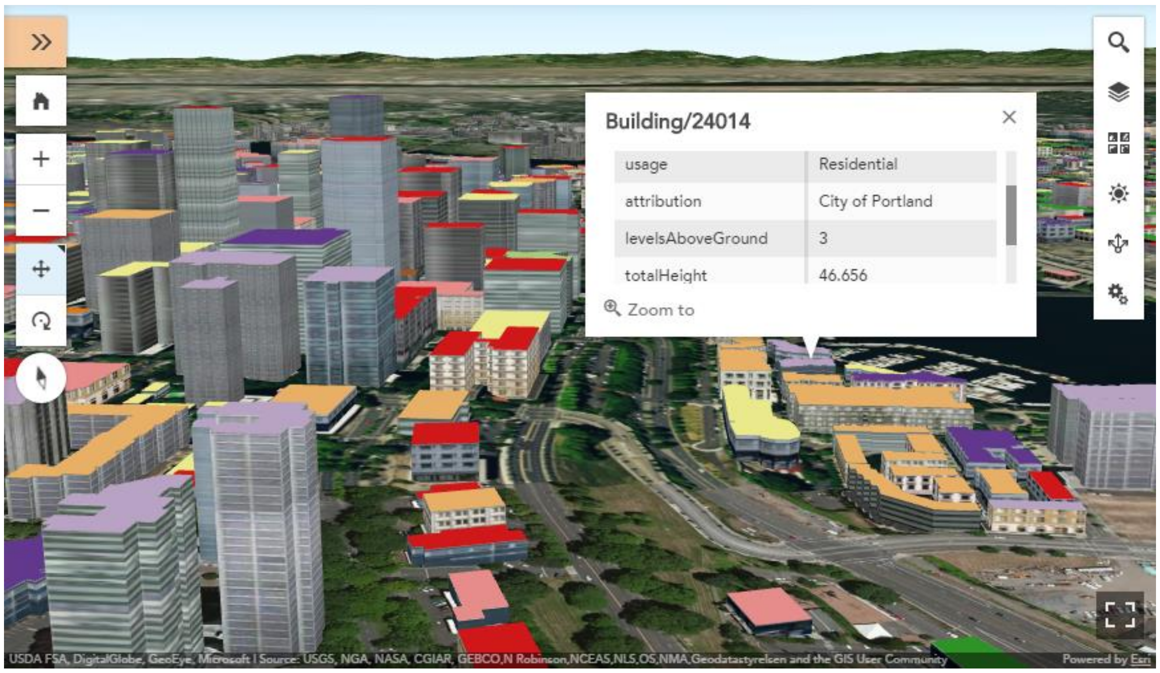
9.6 Attribute Model and Storage

I3S supports the following two patterns of accessing the attribute data:

1. From optional paired services that expose queryable and updatable RESTful endpoints that enable direct access to dynamic source data, including attributes. The query in this case uses the *unique feature-ID* key – which is always maintained within each node and is also available as part of the descriptor for any segmented geometry.

2. From fully cached attribute information, in binary or textual form, as part of the i3S layer.

I3S clients can still choose to access attributes through either of these modes even if the attributes are fully cached within the I3S store.


Building/24014

usage	Residential
attribution	City of Portland
levelsAboveGround	3
totalHeight	46.656

Zoom to
USDA FSA, DigitalGlobe, GeoEye, Microsoft | Source: USGS, NGA, NASA, CGIAR, GEBCO,N Robinson,NCEAS,NLS,OS,NMA,Geodatastyrelsen and the GIS User Community
Powered by Esri

Figure 14: Mesh Pyramid Attributes used in a pop-up

Cached attributes use a binary storage representation based on *Array Buffers* which provide significant performance benefits relative to paired services. The attribute values are stored as a geometry aligned, per field (column), key-value pair arrays.

9.7 Level of Detail – LOD Models, Selection Metrics

The concept of Level of Detail (LOD) is intrinsic to the specification. I3S promotes the concept of discrete levels of details with multiple discrete representations of features *and* nodes, where a more detailed representation of a node and its features fully replaces a coarser representation of the node and the replacement applies across all features or elements covered by the nodes.

9.7.1 Discrete LODs

Scene Layers may include discrete levels of detail that apply to the layer as whole and that generalize or summarize information for the features within a node using a generalization or summarization criterion that is based on the level of detail which corresponds to the depth of the node in the tree. The number of discrete levels of detail for the layer corresponds to the number of levels in the index tree for the scene layer. The I3S level of detail concept is analogous to the level of detail concepts for image pyramids as well as for standard raster and vector tiling schemes.

During navigation and traversal of the I3S tree, clients must decide to either

a) discontinue traversal to node’s descendants if the node is not visible in the current 3D view; or
b) to use/render the data within a node if its quality is appropriate for the current 3D view and discontinue traversal to node’s descendants; or
c) to continue traversal until descendant nodes with better quality are found.

These decisions are made using the advertised values for *LOD selection metrics* that are part of the information payload of the node. The I3S specification supports multiple LOD selection models and metrics. An example LOD selection metric is *the maximum screen size* that the node may occupy before it must be replaced with data from more detailed nodes. This model of discrete LOD rendering is referred to in I3S as *node switching*.

In addition to capturing the LOD model implied by the declared Profile for a layer, the I3S specification also explicitly captures the LOD Switching Model for a layer as described above. I3S Scene Layers also include additional optional metadata on the LOD generation process (e.g. *thinning, clustering, generalization*) as non-actionable (to clients) information that is of interest to some service consumers.

9.7.2 Generation of Levels of Detail for 3D Objects – A specific Example

This section outlines Esri’s implementation of automatic level of detail generation for a Scene Layer representing a collection of 3D GIS features from input data at some chosen resolution or explicitly defined level of detail. The first step is to build the I3S bounding volume tree based on the spatial distribution of the 3D GIS features. Once this has been completed, the generation of the information payloads for interior nodes can proceed:

1. *Leaf-node* payloads capture the original information from 3D GIS features
2. The generation of information in *interior* nodes is based on a bottom up generalization of the feature information in its child nodes similar to the process of building an image-pyramid while factoring in the need of the process to work with both explicit geometry and associated textural representations.

9.8 Delivery via REST or as locally consumable Scene Layer Packages

I3S scene layers can be delivered to web, mobile and desktop clients using a number of different patterns.

Most users will interact with scene layers using applications that access cloud or server based information using RESTful interfaces/services. In these cases the scene layer cache, including the i3S nodes and their payloads, resides on the server and is returned to clients via a RESTful interface that exposes the scene layer, its nodes and their associated resources (geometries, attributes, textures) as web addressable resources. The I3S specification contains a complete description of the web addressable resources and their url scheme.

Some users will also interact with a scene layer delivered to them as a single large Scene Layer Package – this is a single file that packages the complete node tree and its resources into an archive that supports direct access to the individual nodes and resources within it. Scene Layer Packages are part of the current I3S implementation with multiple generators and the ability by clients to consume packages containing hundreds of GB of content.

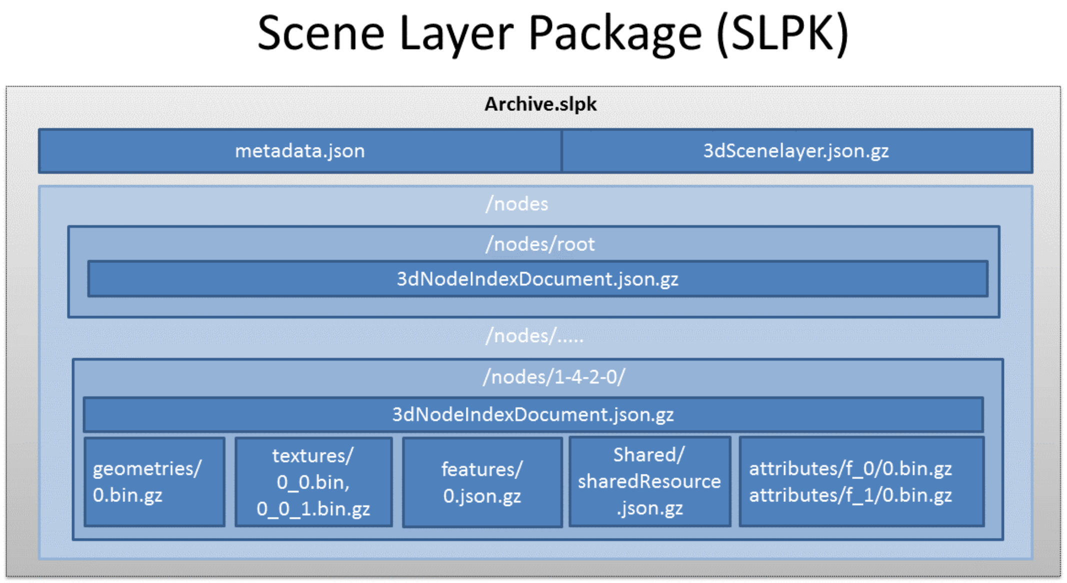

Scene Layer Package (SLPK)
Archive.slpk
metadata.json
3dScenelayer.json.gz
/nodes
/nodes/root
3dNodeIndexDocument.json.gz
/nodes/.....
/nodes/1-4-2-0/
3dNodeIndexDocument.json.gz
geometries/ 0.bin.gz
textures/ 0_0.bin, 0_0_1.bin.gz
features/ 0.json.gz
Shared/ sharedResource .json.gz
attributes/f_0/0.bin.gz attributes/f_1/0.bin.gz

Figure 15: An example of a Scene Layer Package with its structure.

An SLPK is an archive with, typically, a STORE compression schema applied to it. Note also that all SLPK content files, either text (.json), or binary (.bin), are individually compressed with GZIP compression (.gz). Though the SLPK in Figure 9 contains many nodes (as indicated by "/nodes/….."), it shows only node "1-4-2-0" as an example.

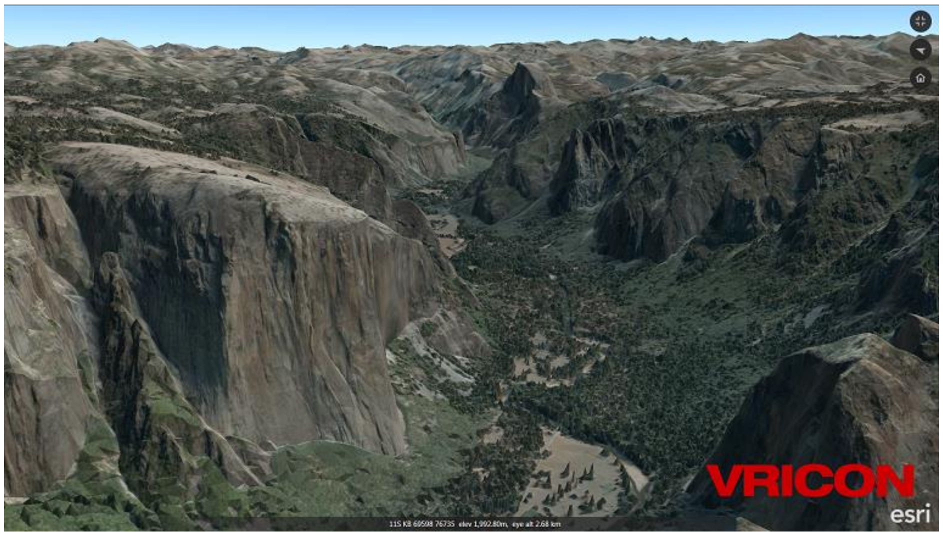


Figure 16: Vricon integrated mesh Scene Layer Package shown in ArcGIS Earth.

9.9 <u>I3S Flexibility</u>

I3S is flexible and allows for different implementation choices for different types of 3D data or even for the same type of 3D data. The profile for a layer indicates the set of choices made. Choices supported by I3D and made use of by different profiles are described below. In each case the profile listed is the canonical profile for the corresponding layer-type.

1. The Minimum Bounding Volume (MBV) property can be represented as:
    a. Minimum Bounding Sphere (MBS)
    b. Oriented Bounding Box (OBB)

2. The node structure may be
    a. 'expanded' – in support of clients that want to gain more complete meta-information about node's position within BVH topology and its immediate neighborhood
        - Each index node provides pointers to its parent, all its children, and sibling neighbors (including their MBVs)
        - Used by: the ***mesh-pyramids*** and ***points*** profiles.
    b. 'fixed-size' – in support of '*paged*' access pattern
        - A minimal structure – just the essentials: bounding volume; first-child reference; child-count; LOD selection data; etc.
        - Used by: the ***pointclouds*** profile.

3. Nodes may have "embedded" vs "binary" geometry content format
    a. Embedded geometry: as text (JSON) inlined with other metadata within *featureData* resource – to support profiles where run-length encoding of feature-IDs along the vertex data is suboptimal – used by: the canonical ***points*** profile.
    b. Binary format: for voluminous, ready to render/use geometries and cached attributes. Both typed array buffer views as well as fixed format binary buffers are supported.
        - The ***mesh-pyramids*** profile uses 'array buffer views' (ArrayBufferView follows the Khronos Typed Array specification)
        - The ***pointclouds*** profile uses little-endian binary buffers in order to support a domain-specific data compression

4. LOD Selection based on different metricTypes:
    1. *maxScreenThreshold* – LOD switching based on screen 'size' of the node's MBV – used by: ***mesh-pyramids*** profile
    2. *screenSpaceRelative* – LOD switching based on screen 'scale' of the node's MBV – used by: ***points*** profile
    3. *distancRangeFromDefaultCamera* – LOD switching based on normalized distance of the node's MBV from the camera – used by: ***points*** profile

effectiveDensity – estimation of the point density covered by the node – used by: ***pointclouds*** profile.

10. Annex B: Additional Examples of Implementation

The following implementations use the proposed Community standard.

Implementation name: ArcGIS Pro 1.4 (prerelease)

Date of most recent version: Prerelease in September 2016; official release planned 2016Q4

Implementation description: ArcGIS Pro implements viewing, sharing, and packaging of scene layers. ArcGIS Pro has implemented 3D Object, 3D Point, and Point Cloud (at 1.4) profiles for viewing, sharing, and packaging. ArcGIS Pro can package scene layers stand alone or it can publish data to ArcGIS for Server which is then converted to scene layers by ArcGIS for Server. ArcGIS Pro can share Integrated Mesh scene layers to ArcGIS for Server or to ArcGIS Online and it can view Integrated Mesh layers.

Any Esri ArcGIS Desktop licensed user can access ArcGIS Pro and use it to create and consume I3S content.

Implementation URL: http://www.esri.com/en/software/arcgis-pro

Is implementation complete? [✓ **Yes**] **No**

If not, what portions of the proposed Community standard are implemented?

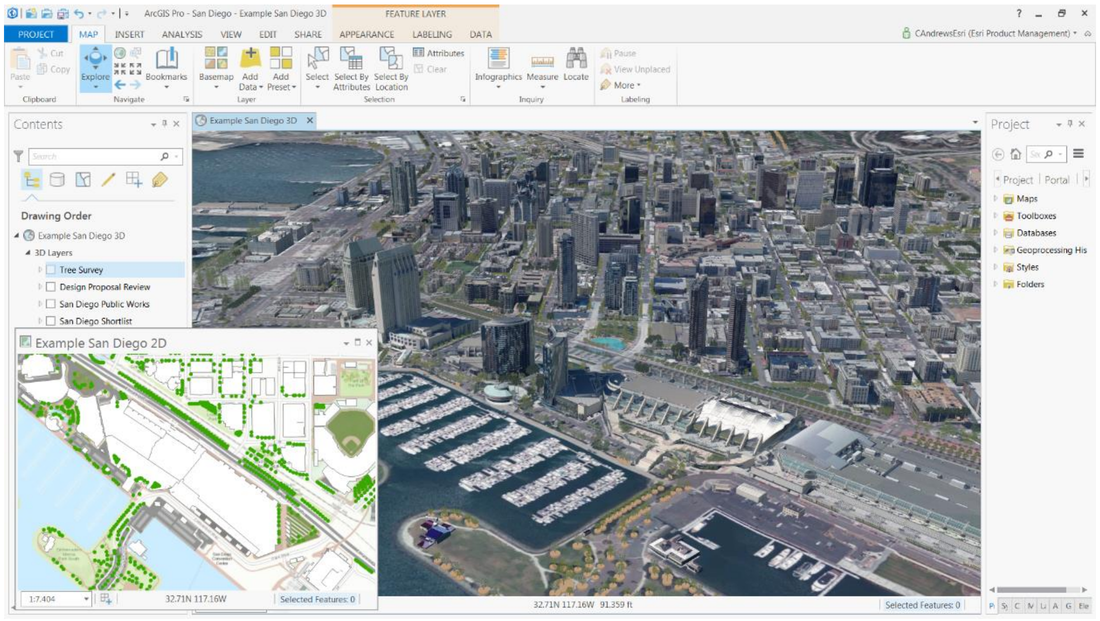


Figure 17. ArcGIS Pro 1.0 showing PLW Modelworks building content cached as I3S with additional GIS data

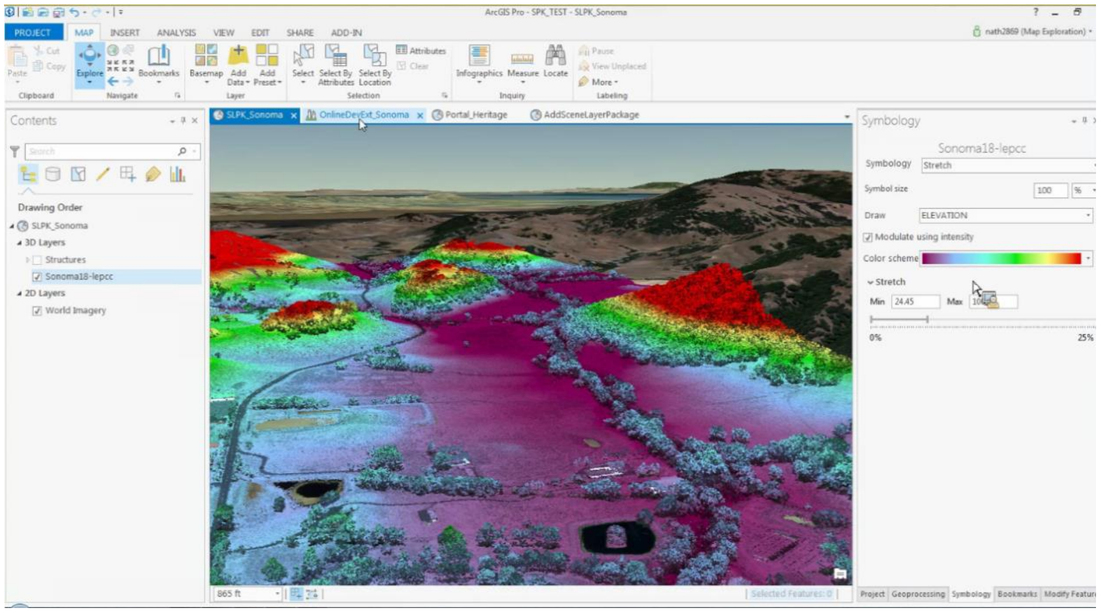


Figure 18. ArcGIS Pro 1.4 prerelease displaying a point clouds scene layer package

Implementation name: ArcGIS Server 10.5

Date of most recent version: Currently in Beta as of August 2016; production release planned in 2016Q4

Implementation description: ArcGIS for Server deploys in both cloud and on premises server infrastructure and can cache, host, and share 3D Point and 3D Object scene layers. ArcGIS for Server can host Point Cloud scene layers and Integrated Mesh scene layers.

Users of client-side components that ship with ArcGIS for Server, including the web scene viewer, Web AppBuilder, and custom web apps built with the Esri JavaScript 4.0 API, may consume and view scene layers from ArcGIS for Server or ArcGIS Online.

Implementation URL: http://www.esri.com/software/arcgis/arcgisserver

Is implementation complete? ☒ **Yes** ☐ **No**

If not, what portions of the proposed Community standard are implemented?

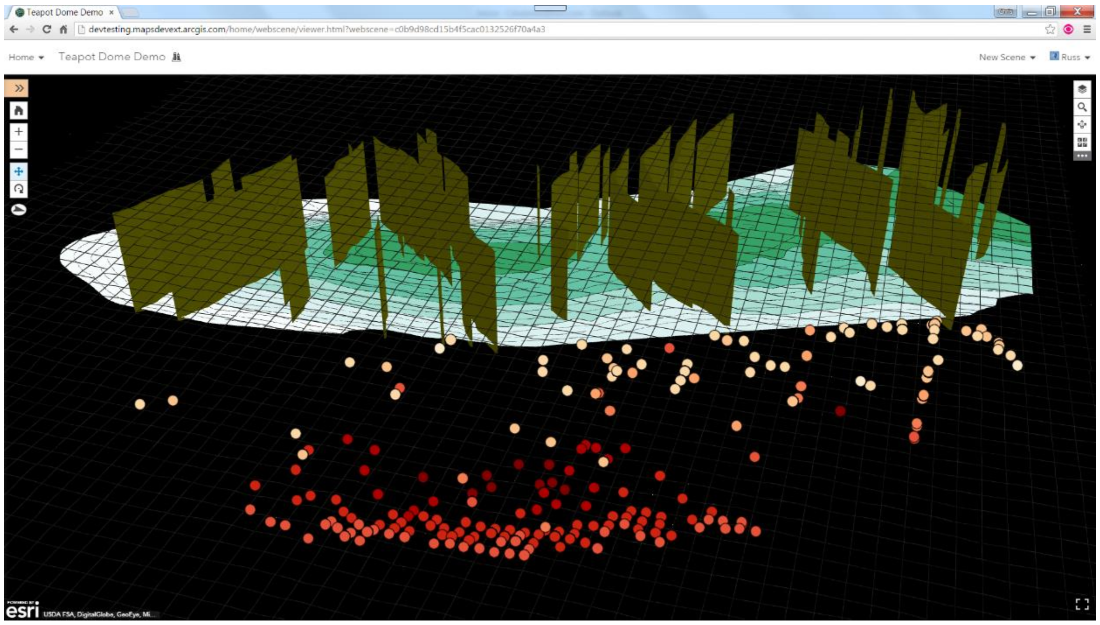

Teapot Dome Demo
devtesting.mapsdevext.arcgis.com/home/webscene/viewer.html?webscene=c0b9d98cd15b4f5cac0132526f70a4a3
Home
Teapot Dome Demo
New Scene
Russ
esri
USDA FSA, DigitalGlobe, GeoEye, MI

Figure 19. Feature services and scene layers combined in a local coordinate system web scene served from ArcGIS Server

Implementation name: ArcGIS Online

Date of most recent version: September 2016

Implementation description: ArcGIS Online, Esri's cloud hosted SaaS environment for managing and sharing geospatial data, can host and share 3D Point, Point Cloud, Integrated Mesh and 3D Object scene layers.

Users of client-side components that ship with ArcGIS for Server, including the web scene viewer, Web AppBuilder, and custom web apps built with the Esri JavaScript 4.0 API, may consume and view scene layers from ArcGIS for Server or ArcGIS Online.

As of September 2016, ArcGIS Online currently hosts thousands of scene layers and scene layer package files.

Implementation URL: http://www.arcgis.com

Is implementation complete? **[✓ Yes]** **No**

If not, what portions of the proposed Community standard are implemented?

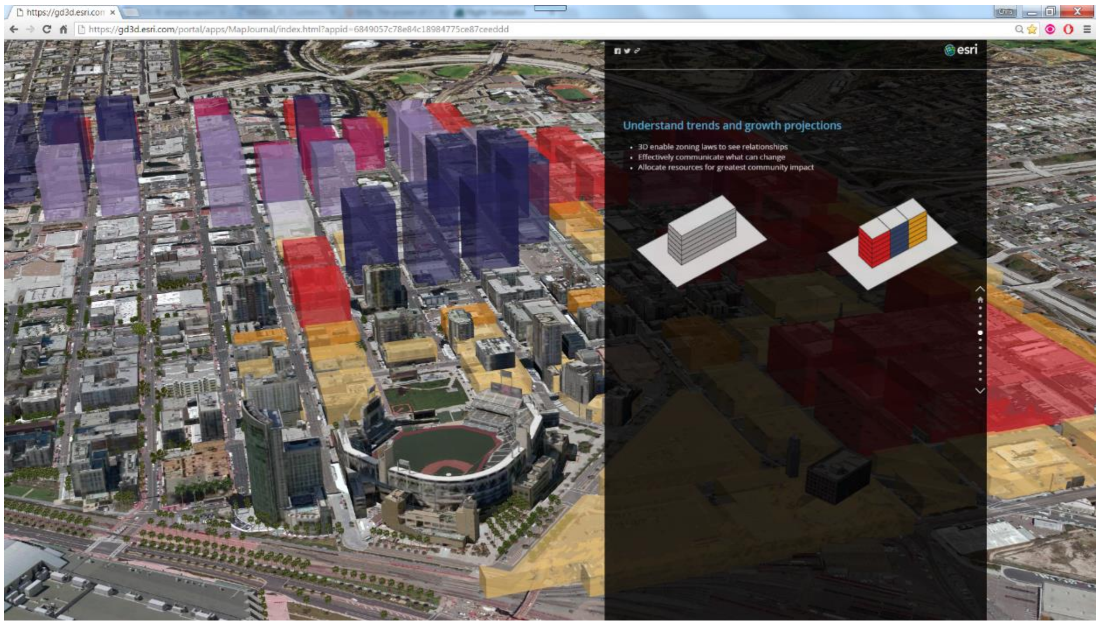

esri
Understand trends and growth projections
3D enable zoning laws to see relationships
Effectively communicate what can change
Allocate resources for greatest community impact

Figure 20. Story Map containing I3S content published in ArcGIS Online in 2015

Implementation name: ArcGIS Earth 1.2.1

Date of most recent version: July 14, 2016

Implementation description: ArcGIS Earth is a standalone desktop client built on Esri's ArcGIS for .NET Runtime SDK. Earth is a viewer and is able to access scene layers served from ArcGIS for Server and ArcGIS Online, including scene layers that are protected behind IWA or PKI credentials.

In addition to scene layers accessed through REST interfaces, ArcGIS Earth is able to directly read scene layers packaged as SLPK files, achieving fast performance with scene layer packages on data sets that are many orders of magnitude larger than traditional data sets. The largest Scene Layer Package that has been tested with ArcGIS Earth is a 120GB data set that is a single file containing Vricon, Inc. integrated mesh i3S data.

ArcGIS Earth has been downloaded over 50,000 times since January 2016.

Implementation URL: http://www.esri.com/earth

Is implementation complete? ☐ **Yes** [✓ **No**]

If not, what portions of the proposed Community standard are implemented?

Viewing integrated mesh and 3D Objects scene layers

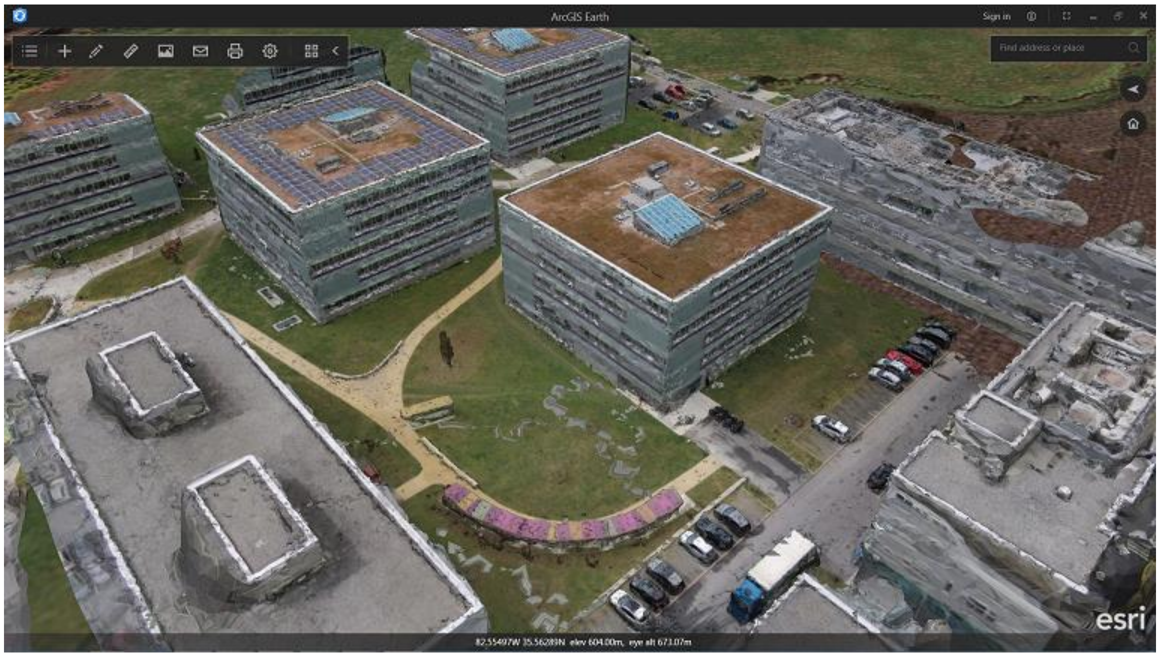


Figure 21: ArcGIS Earth showing an integrated mesh scene layer that was created in Drone2Map and shared using ArcGIS Online.

Implementation name: Esri Drone2Map 1.0

Date of most recent version: June 24, 2016

Implementation description: Drone2Map is a standalone desktop client built on Esri's ArcGIS for .NET Runtime SDK that is used to process drone-captured imagery into i3S layer packages that may be shared in ArcGIS for Server and ArcGIS Online or viewed directly in ArcGIS Earth.

Implementation URL: http://www.esri.com/products/drone2map

Is implementation complete? ☐ **Yes** ☒ **No**

> **If not, what portions of the proposed Community standard are implemented?**
>
> Drone2Map can currently create and view integrated mesh scene layers and scene layer packages.

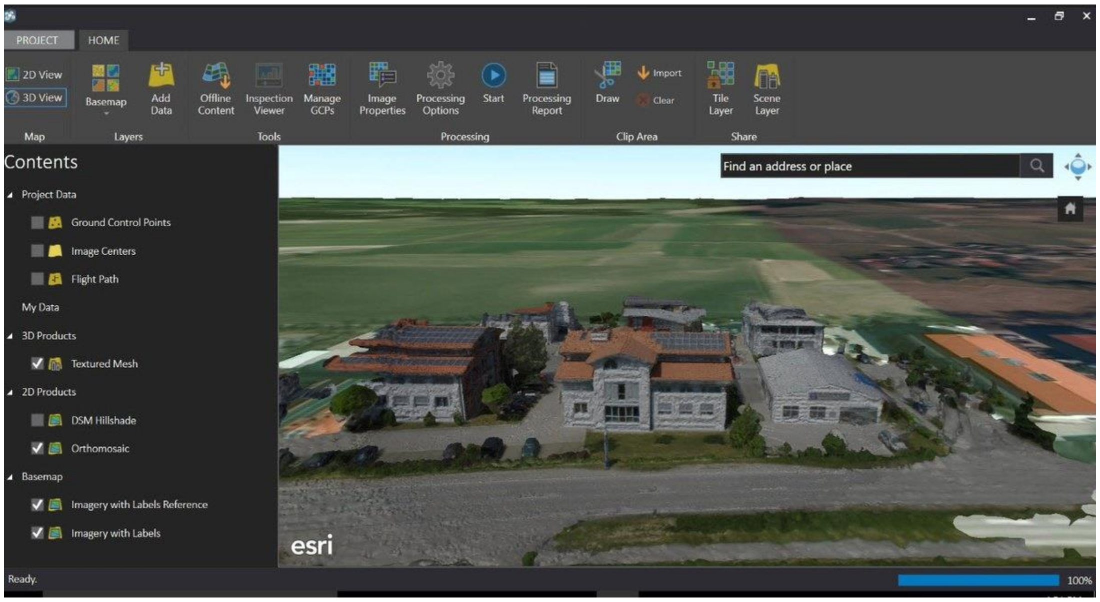


Figure 22. Esri Drone2Map can be used to create I3S integrated mesh content from drone-collected imagery

Implementation name: ArcGIS Runtime SDKs

Date of most recent version: 10.2.7 ArcGIS Runtime SDK for .NET was released February 3, 2016

Implementation description: Built natively from the ground up using C++ and GPU acceleration, ArcGIS Runtime SDKs expose the full capability of the ArcGIS Platform to developers building apps for mobile, desktop, and embedded devices. Runtime apps can consume services and content from ArcGIS Online, ArcGIS Server or run disconnected by accessing local files.

Implementation URL: https://developers.arcgis.com/net/

Is implementation complete? ☐ **Yes** ☒ **No**

If not, what portions of the proposed Community standard are implemented?

The ArcGIS Runtime SDK for .NET 10.2.7 is able to consume integrated mesh and 3D Object scene layers.

The next major version of the ArcGIS Runtime SDKs available for Android, iOS, OS X, .NET, Java, Qt, and Xamarin are available in beta, referred to as “Quartz,” and can consume 3D object, 3D point, point cloud, and integrated mesh scene layer types.

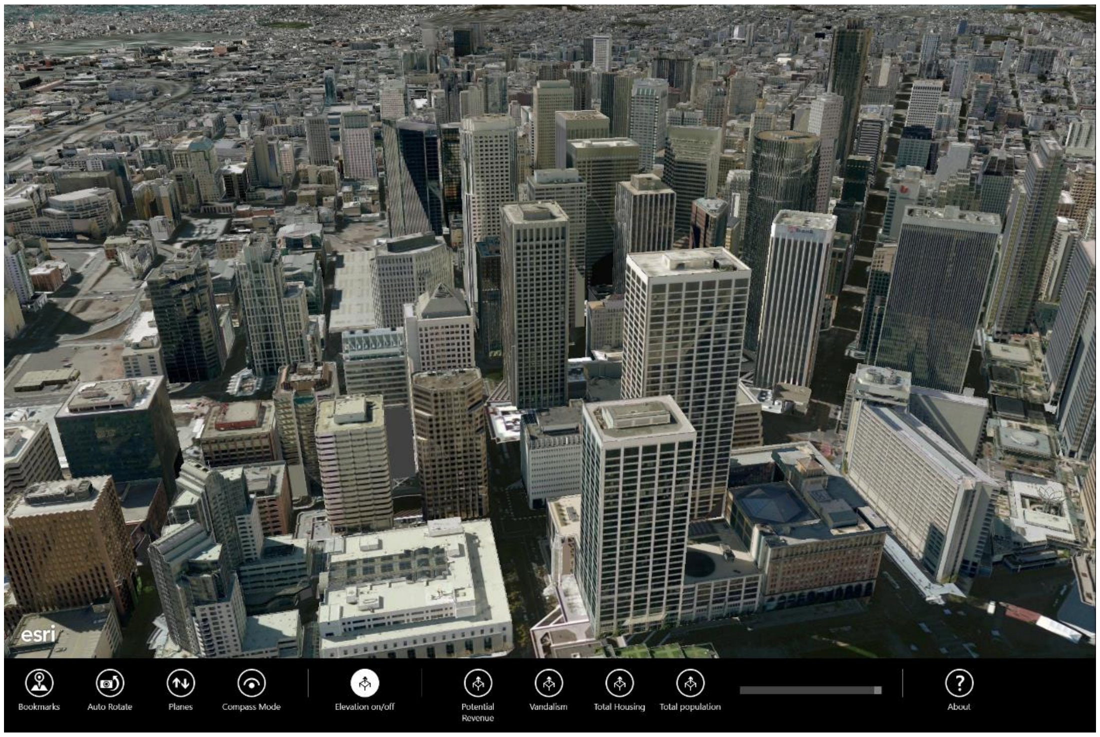
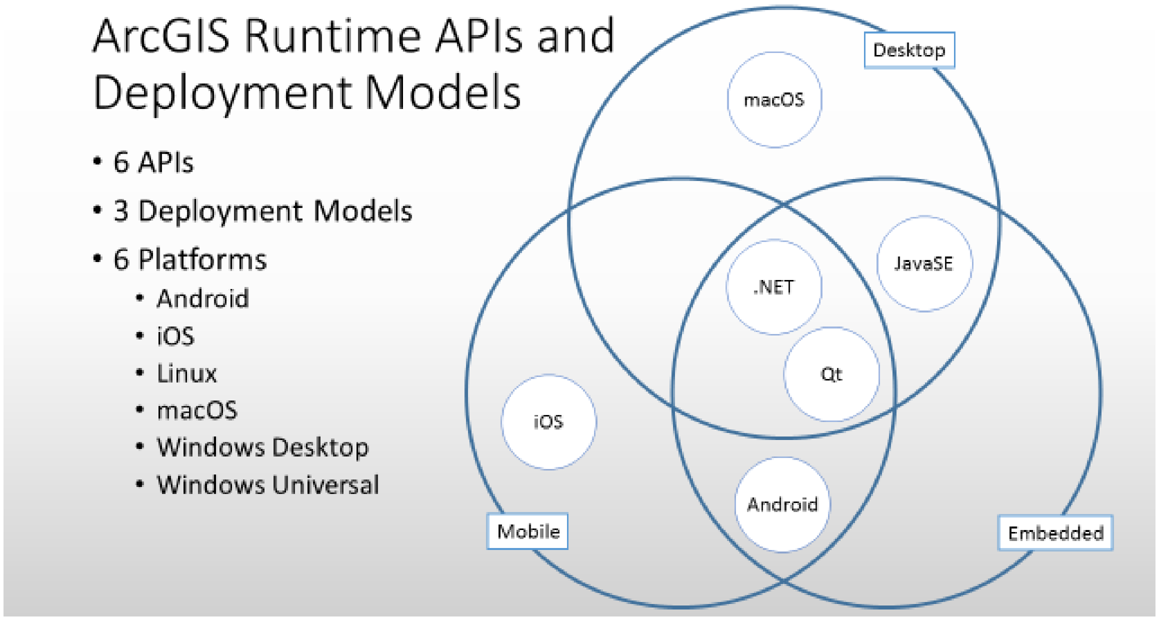


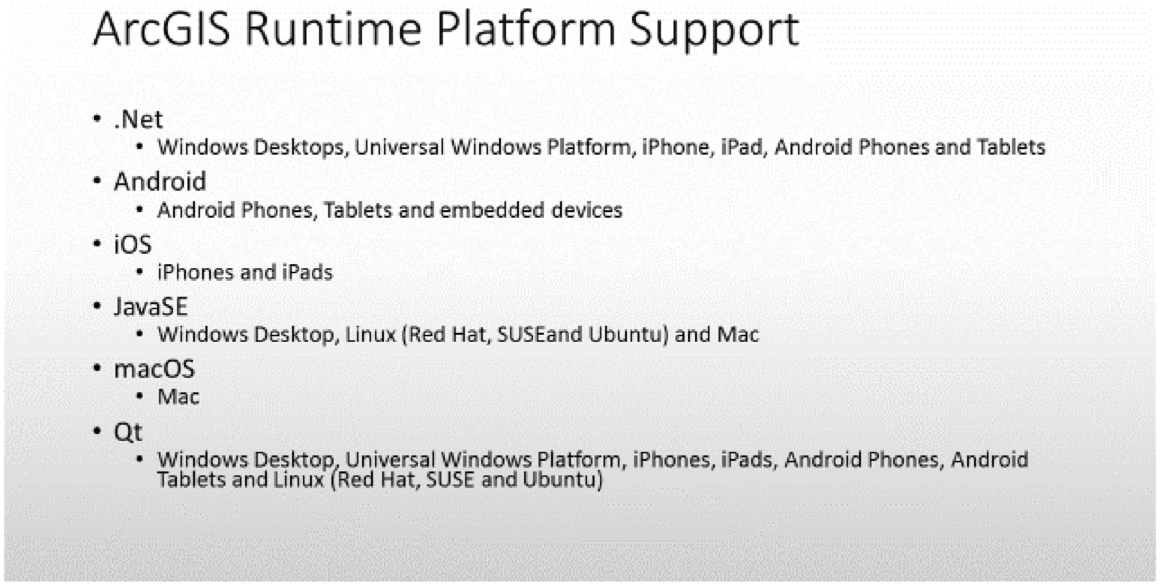
Figure 23. Early example of an ArcGIS Runtime SDK for .NET app accessing scene layers on a Microsoft Windows device



ArcGIS Runtime APIs and Deployment Models

- 6 APIs
- 3 Deployment Models
- 6 Platforms
  - Android
  - iOS
  - Linux
  - macOS
  - Windows Desktop
  - Windows Universal

Figure 24. The ArcGIS Runtime SDKs extend common access to services to app developers across platforms


ArcGIS Runtime Platform Support

- .Net
  - Windows Desktops, Universal Windows Platform, iPhone, iPad, Android Phones and Tablets
- Android
  - Android Phones, Tablets and embedded devices
- iOS
  - iPhones and iPads
- JavaSE
  - Windows Desktop, Linux (Red Hat, SUSEand Ubuntu) and Mac
- macOS
  - Mac
- Qt
  - Windows Desktop, Universal Windows Platform, iPhones, iPads, Android Phones, Android Tablets and Linux (Red Hat, SUSE and Ubuntu)

Figure 25. The ArcGIS Runtime SDKs detailed platform availability

Implementation name: CityEngine

Date of most recent version: 2016.1 – October 2016

Implementation description:

CityEngine supports the creation of Scene Layer Packages (SPLKs) from 3D models. Those models can be imported from 3rd party formats such as FBX, DAE, or OBJ or procedural generated using Computer Generated Architecture (CGA). CityEngine also supports enterprise GIS data management through the navigator. CityEngine users can share scene layer packages by logging into ArcGIS Enterprise or ArcGIS On-Line, uploading the SLPK, and then using the enterprise GIS portal to publish the SLPK as a service.

Implementation URL: http://www.esri.com/software/cityengine

Is implementation complete? No.

If not, what portions of the proposed Community standard are implemented?

CityEngine is currently only producing i3s pyramid mesh SLPKs from model data. These are published without buddy feature layers, so do not currently support attributes. In the future there is planned support for integrated mesh to be exported from street graphs and 3D models. In addition CityEngine will be adding support for the consumption of various i3s services, including Integrated Mesh, Pyramid Mesh, and Points Clouds. No distinct timeline has been set for implementation of consumption at this time.

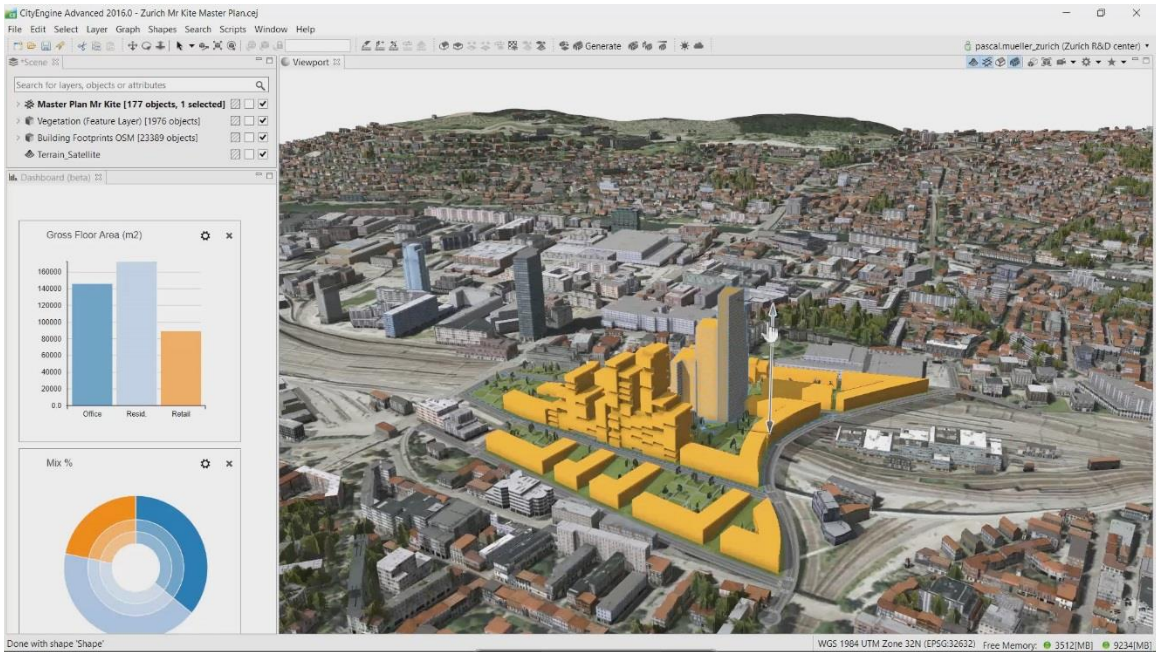

CityEngine Advanced 2016.0 - Zurich Mr Kite Master Plan.cej
File Edit Select Layer Graph Shapes Search Scripts Window Help
Master Plan Mr Kite [177 objects, 1 selected]
Vegetation (Feature Layer) [1976 objects]
Building Footprints OSM [23389 objects]
Terrain_Satellite
Gross Floor Area (m2)
Office
Resid.
Retail
Mix %
Viewport
Done with shape 'Shape'
WGS 1984 UTM Zone 32N (EPSG:32632)

Figure 26. CityEngine shown being used to prepare data for export as a scene layer package file

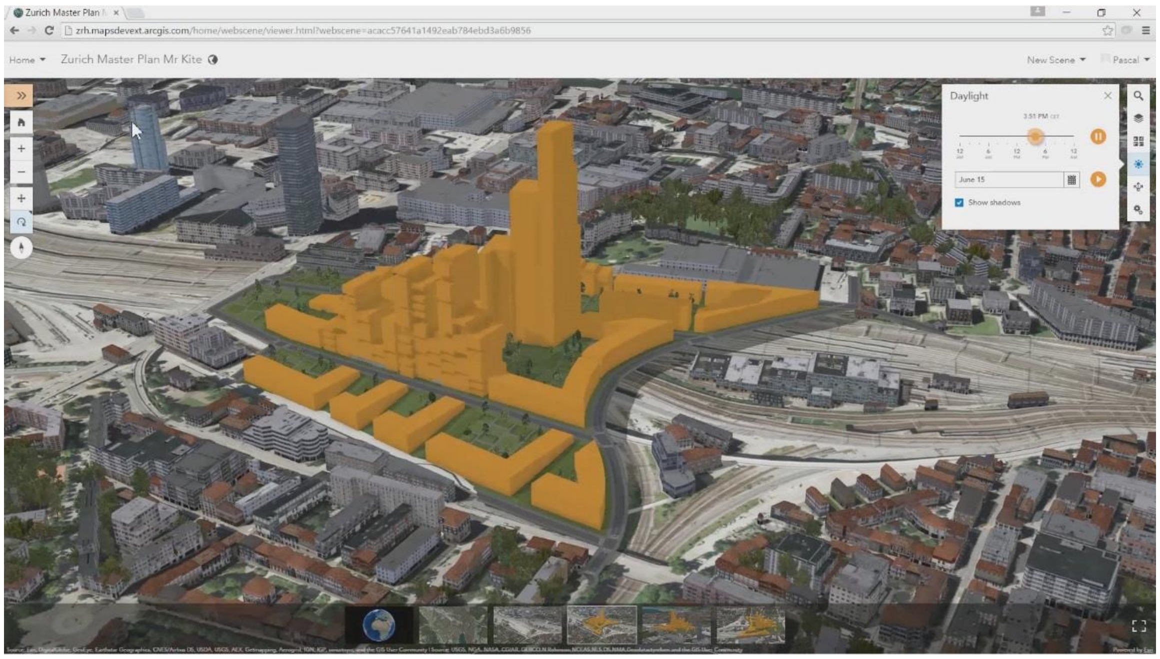

Zurich Master Plan
zrh.mapsdevext.arcgis.com/home/webscene/viewer.html?webscene=acacc57641a1492eab784ebd3a6b9856
Home
Zurich Master Plan Mr Kite
New Scene
Pascal
Daylight
June 15
Show shadows

Figure 27. 3D model from CityEngine shown after being exported as a scene layer package and loaded in a web scene in ArcGIS Online