全空间三维模型数据格式及服务接口规范

M3D 2.2

Specification of data format and service API for pan-spatial 3D model data

目　　次

全空间三维模型数据格式及服务接口规范

1 范围

本文件规定了全空间三维模型数据格式基本规定、文件组织结构及存储格式要求，同时规定了全空间三维模型数据服务接口标准。

本文件适用于网络环境和离线环境下多源异构三维地理空间数据的传输与解析、存储、绘制、发布、共享与互操作，也适用于大场景全空间三维空间数据在不同终端（移动设备、浏览器、桌面电脑）上的三维地理信息系统相关应用。

2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 7408-2005 数据元和交换格式 信息交换 日期和时间表示法

GB/T 16831-2013 基于坐标的地理点位置标准表示法

GB/T 23707-2009 地理信息 空间模式

GB/T 25530-2010 地理信息 服务

GB/T 33187.1-2016 地理信息简单要素访问 第1部分：通用架构

GB/T 35634-2017 公共服务电子地图瓦片数据规范

3 术语和定义

GB/T 20000.1界定的以及下列术语和定义适用于本文件。

3.1

全空间 pan-spatial

数据格式涵盖空中、地上、地表、地下等多种数据类型，覆盖全空间区域，包含但不限于：

——空中：重力场、电磁场、风力场、大气污染指数、噪声污染等数据；

——地上：包括倾斜摄影数据、点云数据等实景三维数据、BIM模型数据；

——地表：包括矢量数据、地形DEM数据；

——地下：包括钻孔、剖面、地质体、网格、属性体等地质数据、地下管线、地下构筑物数据。

3.2

瓦片数据 tile data

根据一定的格网划分规则，对确定地理覆盖范围的地图进行分块形成若干图片单元。

[来源：GB/T 35634—2017,2.1]

3.3

M3D 数据压缩文件 M3D data compressed files

存放M3D数据内容的压缩包，每一个瓦片下面挂接0个或者1个数据包。

3.4

二进制数据流 binary stream

通过二进制数据流来写入和读取数据。

3.5

材质 material

模型对象表面各可视化属性的集合，包括模型对象表面的色彩、纹理、光滑度、透明度、反射率、折射率、发光度等。

3.6

纹理 texture

纹理贴图信息，包含宽、高、压缩方式及纹理二进制数据等。

3.7

三角网几何结构 triangulation geometry

由一系列连续三角形构成的网状平面控制图形，是布设水平控制网的一种形式。

3.8

要素 feature

是M3D中的单个组件，例如3D模型或点云中的点，包含位置、外观和元数据属性等。

3.9

边界范围框 Bounding Volume

完全包含一组几何对象的并集的闭合边界。

4 缩略语

下列缩略语适用于本文件。

glTF：GL传输格式（The GL Transmission Format）
GLB：glTF数据的二进制扩展格式（glTF-Binary）
GLBX：GLB数据的扩展格式（GLB-Extension）
LOD：细节层次（Level of Detail）
M3D：全空间三维模型（Model of 3D）
REST：表现层状态转化（Representational State Transfer）
UML：统一建模语言（Unified Modelling Language）

5 基本规定

5.1 基本数据类型

本文件涉及的基本数据类型规定见表1。

表 1　基本数值类型规定

类型	字节数	取值范围	描述
byte	1	[0,255]	无符号单字节
bool	1	0|1	布尔型
int16	2	[-32768,32767]	短整型
uint16	2	[0,65535]	无符号短整型
int32	4	[-2147483648, 2147483647]	整型
uint32	4	[0,4294967295]	无符号整型
int64	8	[-2^63,(2^63-1)]	长整型
uint64	8	[0,(2^64-1)]	无符号长整型
float	4	[-3.4×10^38,3.4×10^38]	单精度浮点型
double	8	[-1.7×10^{308},1.7×10^308]	双精度浮点型
wchar	2	[-32768,32767]	宽字符类型

5.2 json 格式存储

本文件涉及的json格式存储，规定UTF8编码，不带BOM头。

5.3 服务标准

M3D服务接口遵循RESTful设计规范，可通过桌面端、浏览器端调用该服务。

6　数据结构

6.1 数据文件结构

M3D数据采用数据文件和节点描述文件分离的数据结构，通过文件夹组织数据，可在不加载实际数据的情况下，获取M3D数据节点的关键信息，如包围盒、LOD切换信息以及挂接的数据文件等内容。M3D数据文件提供两种结构，即M3D属性外置结构和M3D属性内嵌结构，可以根据数据的大小和访问频率选择合适的数据结构。

6.1.1.　M3D 属性外置结构

M3D属性外置结构适用于浏览数据，较少查看属性的场景，将属性文件与数据文件同级存储，可以加快数据的传输速度，提高浏览效率。

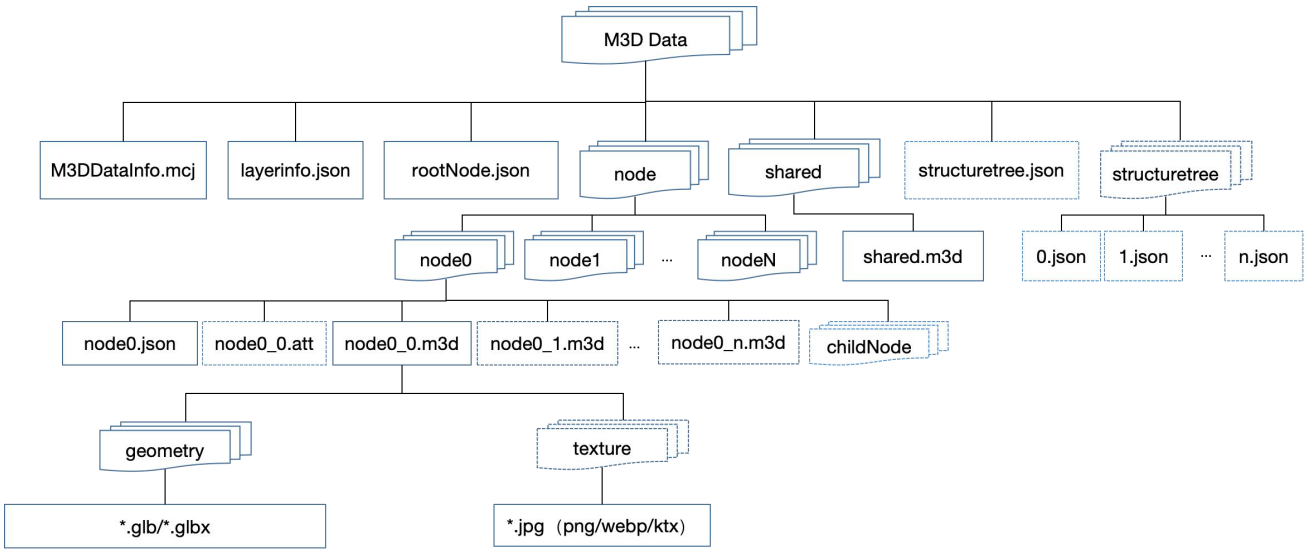


图 1 M3D 属性外置结构设计图

M3D属性外置结构特点：

属性与数据文件同级存储：将属性文件（如描述数据特性的元数据）外置，与实际数据文件存储在相同的文件夹级别，但两者是分开的文件。

加快数据传输速度：由于属性文件和数据文件是分开的，并且属性文件较小，可以更快地加载到内存中，加快数据的传输速度。

提高浏览效率：在不需要频繁访问属性数据的情况下，通过读取节点描述文件快速了解数据节点的结构和位置，提高浏览数据的效率。

6.1.2. M3D 属性内嵌结构

M3D属性内嵌结构适用于经常查询属性的场景，将属性存放在M3D数据文件中，可以减少请求批次，提高浏览效率。

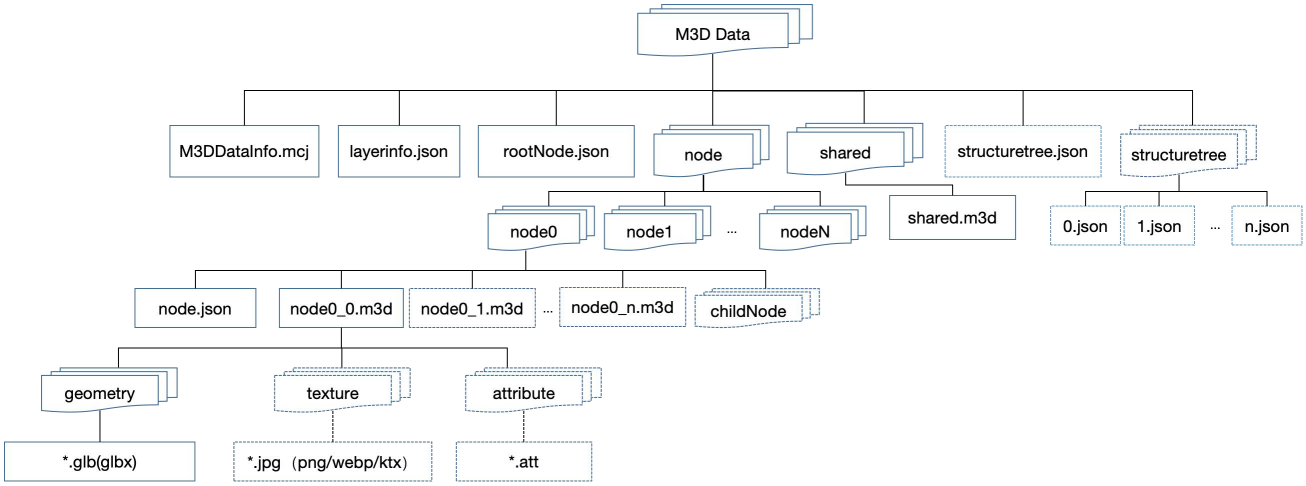


图 2　M3D 属性内嵌结构设计图

M3D属性内嵌结构特点：

属性文件存放在M3D数据文件中：与M3D属性外置结构不同，属性数据被内嵌到M3D文件内部，读取属性数据时不需要额外的文件请求，从而减少网络延迟和I/O操作。

减少请求批次：将属性数据内嵌到M3D数据文件中，减少向服务器发送的请求数量，提高整体性能。

提高浏览和查询效率：在需要频繁访问属性数据的场景中，直接从M3D文件中读取属性数据，而无需额外的文件加载，显著提高浏览和查询的效率。

6.2 数据树形结构

全空间三维模型M3D使用树形结构管理和渲染大规模的三维数据集，采用四叉树、八叉树、K-D树或R树等空间分割方式，高效地处理全空间三维模型数据。

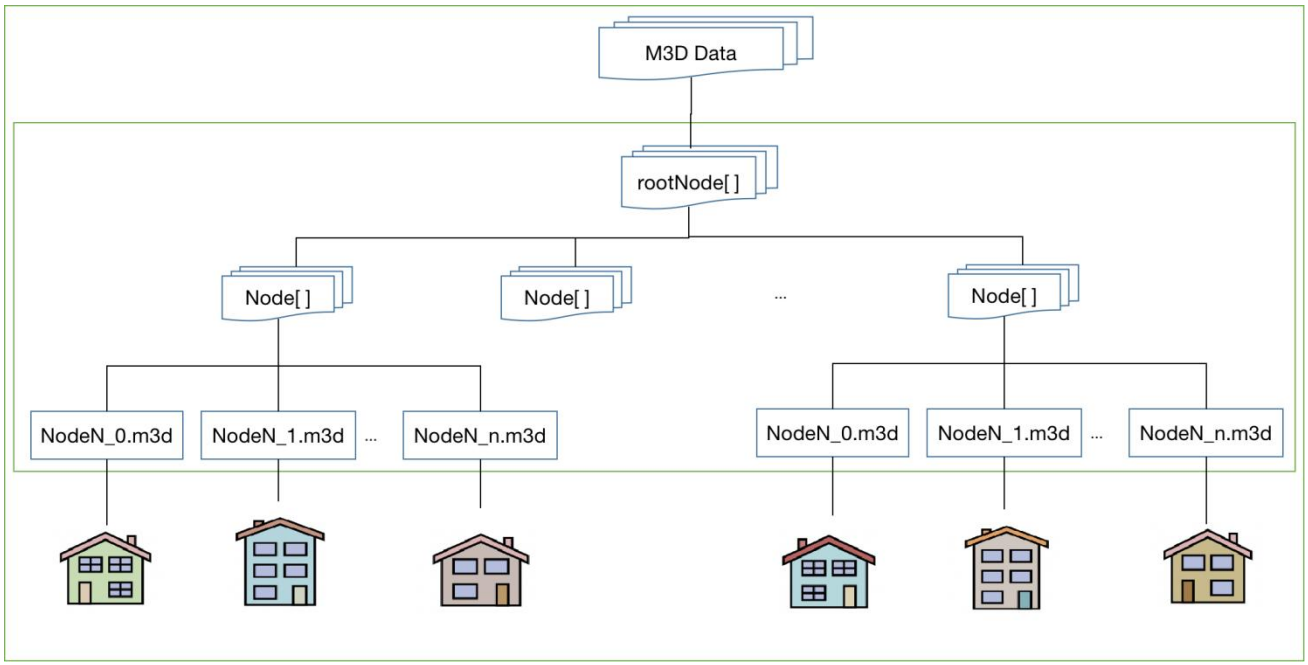


图 3　M3D 数据树形结构图

M3D Data：

M3D 全空间三维数据集，包含和组织所有的 M3D 数据，可代表不同的三维模型、场景或数据块。

rootNode：

将 M3D 数据节点进一步细分为节点，划分为更小的、可管理的部分，便于数据的加载、渲染和传输。

Node[ ]：

在 rootNode 下面存在多个 Node 节点，代表不同的空间区域或数据层，每个 Node 节点包含多个子 Tiles 或更下层的 Node 节点。

NodeN_n.m3d：

每个 Node 节点下有具体的 m3d 文件，用于存储三维数据，包含几何数据、纹理以及其他相关的属性信息。

6.3 数据文件组成

全空间三维数据模型M3D是一个文件系统，包含模型缓存信息描述文件、根节点信息描述文件、图层属性信息描述文件以及节点文件夹等，用于高效管理和渲染大规模的三维数据，可以按需加载和渲染场景的不同部分，也便于对场景进行修改和更新。文件组织结构如下：

+-- M3DDataInfo.mcj（模型缓存信息描述文件）

+-- rootNode.json（根节点信息描述文件）

+-- layerinfo.json（图层属性信息描述文件）

+-- node（节点文件夹）

| +-- 0.json（节点信息描述文件）

| +-- 0.m3d（节点数据）

| +-- 0.att（节点属性数据）

+-- structuretree.json（结构树描述文件）

+-- structuretree（结构树文件夹）

| +-- 0.json（子结构树描述文件）

+-- shared.m3d（公共数据文件）

文件描述见表2。

表 2 文件描述表

文件名	文件类型	文件描述	是否可选
M3DDataInfo.mcj	模型缓存信息描述文件	全空间模型 M3D 缓存数据的信息描述文件，json 格式，后缀名为.mcj，文件名可自定义	必备
rootNode.json	根节点信息描述文件	根节点的信息描述文件，json 格式，后缀名为.json，文件名可以自定义	必备
node.m3d	节点数据文件	节点数据文件，m3d 格式，后缀名为.m3d，文件名可以自定义	必备
layerinfo.json	图层属性信息描述文件	描述图层的属性信息文件，json 格式，后缀名为. json，文件名可以自定义	可选
node.att	节点属性数据文件	节点属性数据文件，存储属性信息，后缀名为.att，文件名可以自定义	可选
structuretree.json	结构树文件	结构树文件，存储结构信息，json 格式，后缀名为. json，文件名可以自定义	可选
shared.m3d	公共数据文件	公共数据文件，m3d 格式，后缀名为.m3d，文件名可以自定义	可选

7　存储内容

7.1 数据空间范围

全空间三维数据模型M3D支持不同类型的数据空间范围（BoundingVolume），包括BoundingBox对象和BoundingSphere对象，两者应二选一，如表3所示。BoundingBox对象和BoundingSphere对象说明见表4、表5。

表 3　BoundingVolume 对象说明

标签名	类型	描述
boundingBox	BoundingBox	数据外包盒，外包盒和外包球二选一
boundingSphere	BoundingSphere	数据外包球，外包盒和外包球二选一

BoundingBox（外包盒）对象是一个由6元素的AABB包围盒，即经纬高三轴的最值：

表 4　BoundingBox 对象说明

标签名	类型	描述
left	double	数据外包盒左边界的经度值
top	double	数据外包盒上边界的纬度值
right	double	数据外包盒右边界的经度值
bottom	double	数据外包盒下边界的纬度值
minHeight	double	数据外包盒高度的最小值
maxHeight	double	数据外包盒高度的最大值

BoundingSphere（外包球）对象是一组四个数字，用于定义边界球体，前三个元素定义球体中心的x、y和z值，最后一个元素定义为半径：

表 5　BoundingSphere 对象说明

标签名	类型	描述
center	Point	包围球中心点，包含 x、y 和 z 值
radius	double	包围球半径

7.2 缓存信息描述文件

7.2.1. 缓存信息描述文件概述

缓存信息描述文件即M3DDataInfo.mcj，用于描述全空间模型M3D缓存数据的基本信息，关联对象的组织结构如下图所示，缓存信息描述文件示例参见附录A.1。

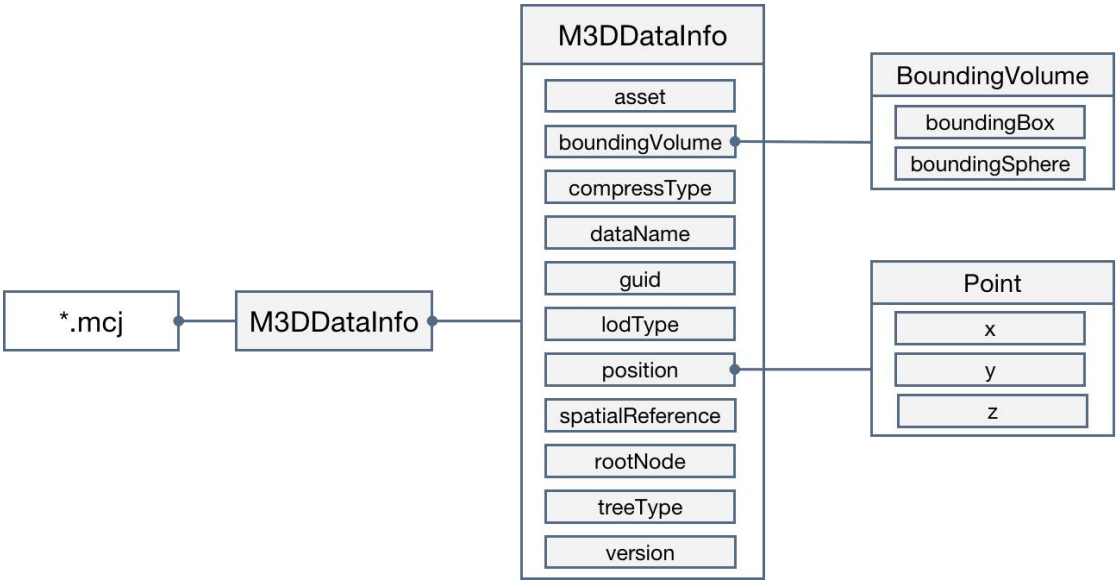


图 4　M3DDataInfo 文件类图

7.2.2.　缓存信息描述文件标签信息

缓存信息描述文件应对整个数据对象进行描述和定义。缓存信息描述文件对象说明见表6、Point对象说明见表7、Uri对象说明见表8。

表 6　缓存信息描述文件对象说明

标签名	类型	描述
asset	string	数据基本信息，如数据所有者
version	string	版本号
dataName	string	数据名称
guid	string	数据唯一标识符
compressType	string	数据压缩类型，取值范围｛“zip”，“7z”，“rar”｝
spatialReference	string	空间参考坐标系信息，取值范围｛“WGS84”｝
treeType	string	树形组织结构类型，取值范围｛“QuadTree”，“OCTree”，“K-DTree”，“RTree”｝，分别表示：四叉树、八叉树、K-D 树、R 树。
lodType	string	LOD 类型，取值范围｛“ADD”，“REPLACE”｝，分别表示添加层次细节与替换层次细节
boundingVolume	BoundingVolume	数据的外包范围，使用外包球或外包盒描述
position	Point	经纬度及高程定位点信息
rootNode	Uri	根节点描述文件数据路径

表 7 Point 对象说明

标签名	类型	描述
x	double	空间点 X 坐标值

y	double	空间点 Y 坐标值
z	double	空间点 Z 坐标值

表 8　Uri 对象说明

标签名	类型	描述
uri	string	文件数据路径

7.3 节点文件

7.3.1. 节点概述

节点信息描述文件（NodeInfo对象），用于描述M3D数据的树形组织结构。

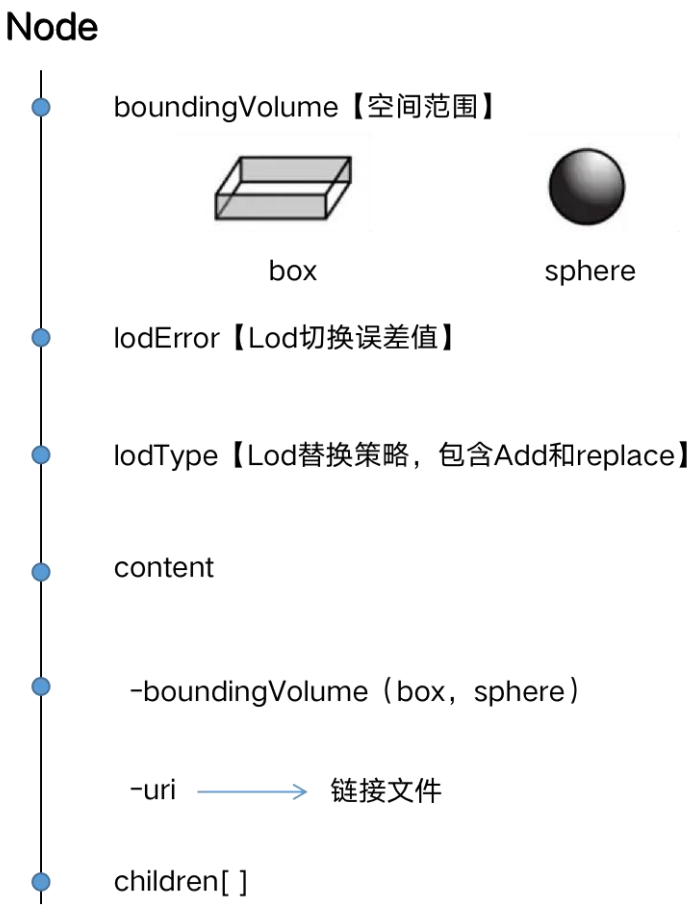


图 7 节点信息组成概念图

boundingVolume（包围盒）：

用于近似表示一个三维模型或一组三维模型的空间范围，常见的包围盒有轴对齐包围盒（AABB）和方向包围盒（OBB）。

LodError（LOD 切换误差值）：

用于控制 LOD 切换精度的阈值。决定在不同距离或像素精度下，应该加载并渲染哪个精细度的模型。通过调整 LodError 的值，可以平衡渲染的质量和性能。LodError 值越大，可以在较远的距离下看到更粗糙的模型，减少渲染负担；LodError 值越小，需要靠近模型才能看到更精细的模型。

LodType（LOD 替换策略）：

描述了不同 LOD 级别模型之间的替换方式，提供 Replace 和 Add 两种策略。

Replace 是在加载新的更精细的 LOD 级别模型时，替换掉旧的粗糙模型，确保在任何时候都只看到一种精细度的模型。

Add 是在加载新的更精细的 LOD 级别模型时，不替换掉旧的模型，而是将新的模型添加到场景中，实现更丰富的视觉效果。

content 内容：

包含了瓦片的范围信息和瓦片文件的 URI（统一资源标识符），可以根据 URI 找到具体的瓦片文件，并加载到场景中进行渲染。瓦片范围信息则用于确定瓦片在空间中的位置和大小。

children 子节点：

表示当前瓦片的子瓦片。每个子瓦片是一个 Tile 对象，具有与父瓦片相同的属性和结构，使用嵌套的方式构建一个 HLOD（Hierarchical Level of Detail）的树形结构，用于高效地管理大量数据，在渲染中可以快速的视锥裁剪和 LOD 切换。

7.3.2. （根）节点信息描述文件

（根）节点信息描述文件（rootNodeInfo对象），用于描述M3D数据的树形组织结构。关联对象的类图如下图，节点信息描述文件示例参见附录A.2。

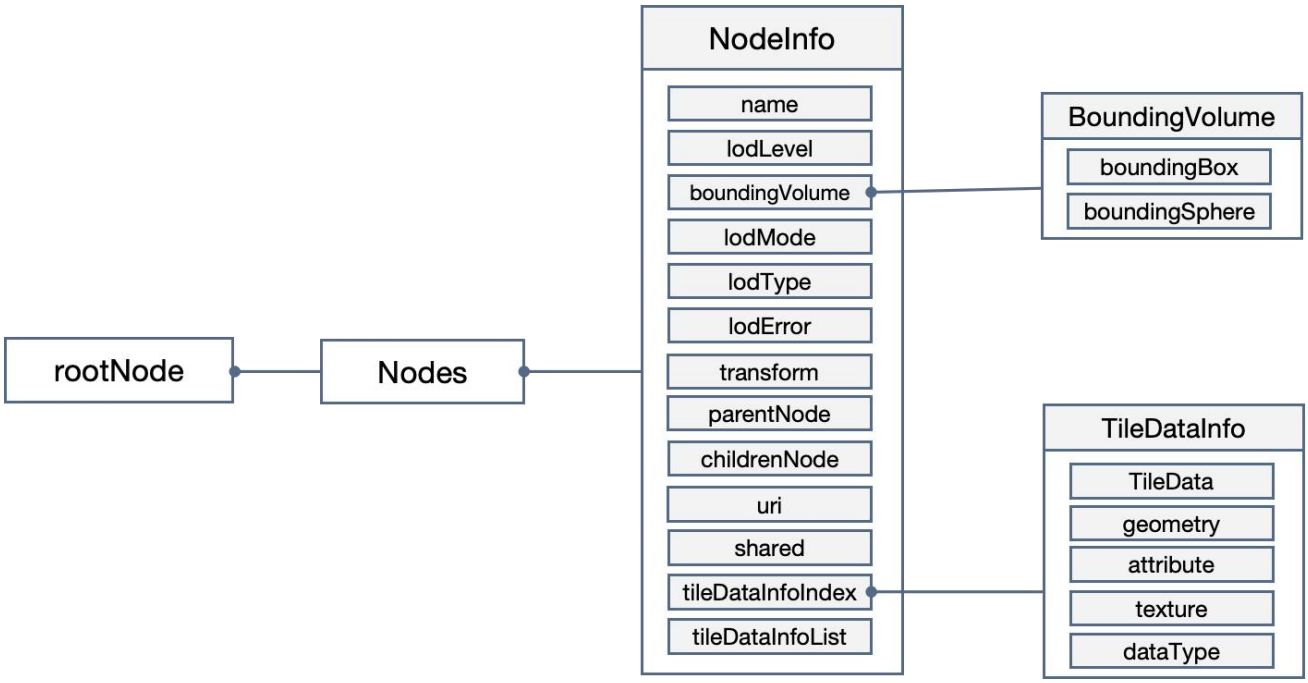


图 8　rootNodeInfo 文件类图

7.3.3. 节点信息描述文件标签信息

节点信息描述文件对象说明见表9、TileDataInfo对象说明见表10。

表 9　（根）节点描述文件对象说明

标签名	类型	描述
name	string	节点名称
lodLevel	string	节点 LOD 级别
boundingVolume	BoundingVolume	节点外包范围
lodType	string	LOD 类型，取值范围｛“ADD”，“REPLACE”｝，分别表示添加层次细节与替换层次细节

lodError	float	最大几何误差，单位为米
transform	float[16]	节点相对转换矩阵，用于描述该节点的相对坐标信息
parentNode	NodeInfo	父节点描述信息，使用 NodeInfo 对象表示
childrenNode	Array<NodeInfo>	子节点描述信息，使用 NodeInfo 数组对象表示
shared	Uri	公共数据路径
tileDataInfoIndex	int	M3D 瓦片数据索引
tileDataInfoList	Array<TileDataInfo>	M3D 瓦片数据列表

表 10　TileDataInfo 对象说明

标签名	类型	描述
tileData	Uri	M3D 瓦片数据
geometry	Geometry	几何数据结构
attribute	Attribute	属性数据结构
texture	Uri	纹理图片数据路径
dataType	string	数据类型，取值范围｛“Vector”，“TiltPhotography”，“Model”，“BIM”，“PointCloud”，“PipeLine”，“GeoModel”“GeoGrid”，“GeoDrill”，“GeoSection”｝，分别表示：矢量、倾斜、模型、BIM、点云、管线、地质体、地质体网格、地质钻孔、地质剖面

7.3.4.　节点数据文件

节点数据文件是以M3D数据压缩文件存储的，存储的是实际的M3D数据信息，包含几何要素文件、属性记录文件和纹理图片文件三部分内容。几何要素文件描述一个空间范围内的三维数据的几何信息；属性记录文件描述数据的属性结构、属性字段和属性记录；纹理图片文件是指材质信息中关联的纹理图片文件。

M3D数据压缩文件的组成结构如下图所示，文件描述如下表。

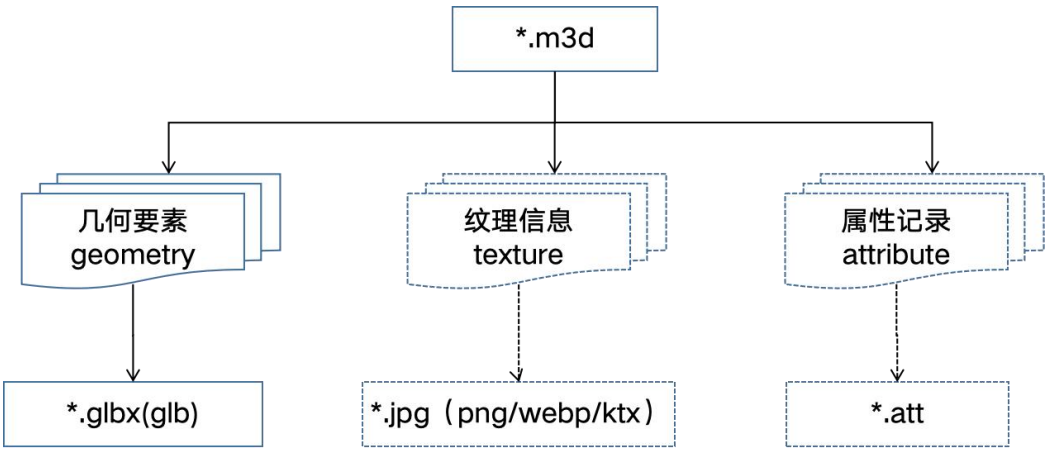


图9 M3D压缩文件组成

M3D数据压缩文件描述见下表所示：

表 11 M3D 数据压缩文件描述

文件名	文件类型	文件描述
*.glb(glbx/pnts/i3dm/cmpt/b3dm/tid)	几何要素	M3D 压缩文件中的几何要素文件，可采用 glTF 数据的二进制扩展格式，后缀名为.glbx，文件名可自定义（基于 glb 数据扩展了单体化信息及地质体数据信息的几何要素文件，后缀名为.glbx，文件名可自定义）
*.att	属性记录	M3D 压缩文件中的属性数据信息，后缀名为.att，文件名可自定义

*.jpg(png/webp/ktx)	纹理信息	M3D 压缩文件中的材质信息中关联的纹理图片文件，后缀名为.jpg，也可以是 png、webp 或者 ktx，文件名可自定义
*.tid	全局编码信息文件	M3D 压缩文件中的全局编码信息，后缀名为.tid，文件名可自定义

7.3.4.1. 几何要素文件

几何要素文件几何要素文件用于描述特定空间范围内三维数据的几何特征，包括空中、地上、地表、地下数据的几何形态及相关数据信息。

1.三角网格几何结构

一般数据均采用三角几何结构表示，其几何描述信息对象说明见下表。

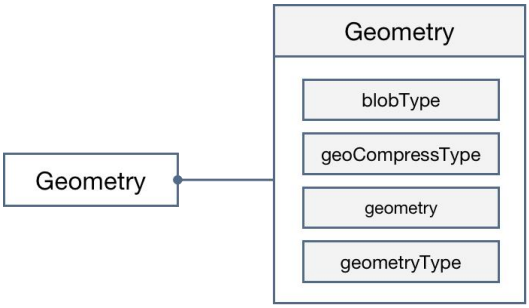


图 10　Geometry 文件组成类图

表 12　几何描述信息对象说明

标签名	类型	描述
blobType	string	几何数据的二进制类型，取值范围｛“glb”，“glbx”｝
geoCompressType	string	几何数据的压缩类型，可选属性，取值范围｛“draco”，“meshopt”｝
geometry	Uri	几何数据路径
geometryType	string	几何类型，取值范围｛“Point”，“Line”，“Polygon”，“Surface”，“Entity”｝

2.全局编码信息

全局编码信息，表示构件树的全局编码信息，采用*.tid存储，由FileHead、tilesOffset[]以及Tiles三部分组成。

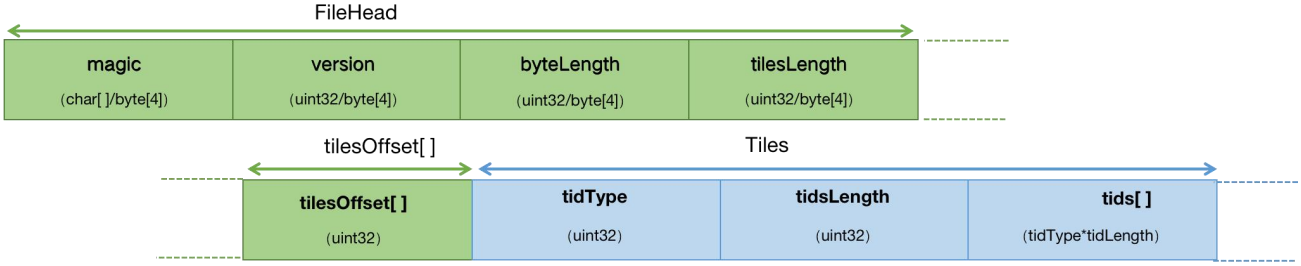


图 9 TID 全局编码组成

(1)FileHead

FileHead是文件的头信息，其内容组织如下表所示：

表 13　FileHead 标签对象信息

标签名	类型	描述
magic	char[]	文件类型，固定取值“tid\0”
version	uint32	版本，值为 1
byteLength	uint32	整个文件的大小，byte 为单位
tilesLength	uint32	瓦片个数

(2)tilesOffset[ ]

tilesOffset[ ]用于记录瓦片的偏移量，其内容组织如下：

表 14 tilesOffset 标签对象信息

标签名	类型	描述
tilesOffset	uint32	记录每个瓦片的偏移量

(3)tiles

tiles是具体的瓦片，其内容组织如下所示：

表 15 tiles 标签对象信息

标签名	类型	描述
tidType	uint32	字段类型，取值范围：｛‘uint16’，‘uint32’，‘uint64’｝
tidLength	uint32	表示该瓦片中有多少个 id
Tids[ ]	uint32	具体的 id 值，TID 数组的下标索引对应于顶点属性

2.地质模型几何结构

地质模型主要涵盖钻孔模型、剖面模型、地质体模型和网格模型，这些地质模型的几何结构信息应存储到glBX几何要素文件中。

（1）钻孔模型几何结构

钻孔模型几何结构上记录上下多层钻孔点与半径，钻孔模型各属性含义见表16。

表 16 钻孔模型各对象说明

属性名	类型	描述
drillPnts	Array<Point>	钻孔点序列数组
drillRadius	double	钻孔柱半径

（2）地质剖面模型几何结构

地质剖面几何结构上记录封闭三维线，剖面模型属性含义见表17。

表 17　剖面模型各对象说明

属性名	类型	描述

sectionLine	Array<Point>	剖面封闭三维线

（3）地质体模型几何结构

地质面模型几何结构用三角网描述，地质体模型由多个地质面构成。地质体对象说明见表18、地质面对象说明见表19。

表 18 地质体对象说明

属性名	类型	描述
geoEntity	Array<GeoSurface>	地质体

表 19 地质面对象说明

属性名	类型	描述
geoSurface	Array<Point>	三角网顶点序列数组
	Array<int>	三角网索引序列数组

3.网格模型几何结构

网格模型包括规则六面体模型和角点网格模型。规则六面体模型几何结构记录几何参数、规则六面体网格序列集合以及属性信息；角点网格模型几何结构由外表面三角网和内部规则格网两部分构成，记录几何参数、规则六面体网格序列集合、外表面点坐标、外表面三角网以及其属性信息。网格模型组成结构见下图。

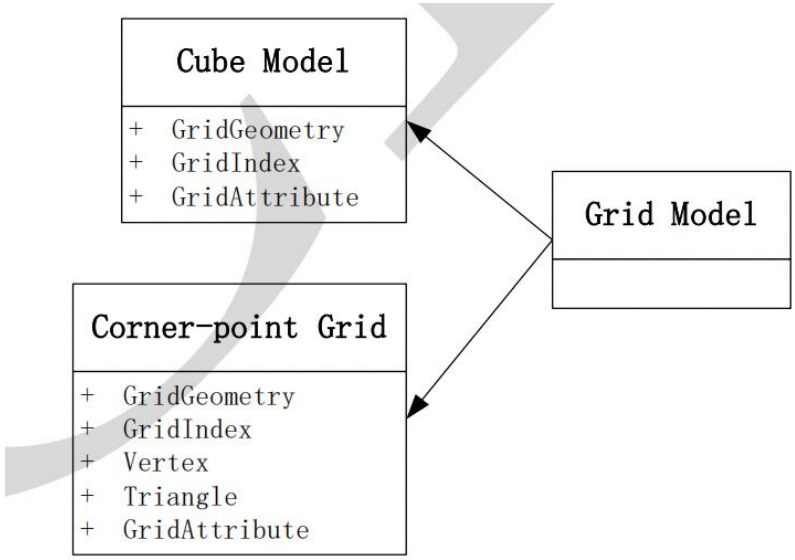


图 12 网格模型几何结构组成

网格模型L_ Point对象说明见GridGeometry数据对象说明见表21、GridIndex数据对象说明见表22、GridAttribute数据对象说明、Vertex数据对象说明及Triangle数据对象说明见下表。

表 20 L_ Point 对象说明

标签名	类型	描述
ix	long	X 方向数量

iy	long	Y 方向数量
iz	long	Z 方向数量

表 21 GridGeometry 数据对象说明

属性名	类型	描述
startPoint	Point	格网原点（x,y,z）
gridStep	Point	格网间距（dx,dy,dz）
gridNum	L_ Point	格网数量（ix,iy,iz）
lodLevel	int	Lod 等级（L）

表 22　GridIndex 数据对象说明

属性名	类型	描述
gridIndexSetNum	long	格网个数
gridIndex	L_ Point	格网序列值
gridIndexSet	Array< L_ Point >	格网序列值集合

表 23　GridAttribute 数据对象说明

属性名	类型	描述
attNum	long	多属性个数（N）
attributes_1	double	Lod 为 1 时所有格网属性 1 的值
attributes_1	double	…
attributes_1	double	Lod 为 L 时所有格网属性 1 的值
…		…
attributes_N	double	Lod 为 1 时所有格网属性 N 的值
attributes_N	double	…
attributes_N	double	Lod 为 L 时所有格网属性 N 的值

表 24　Vertex 数据对象说明

属性名	类型	描述
vertexNum	long	顶点个数
vertexPoint	Point	顶点坐标值
vertexPointSets	Array< Point >	顶点 XYZ 坐标集合

表 25　Triangle 数据对象说明

属性名	类型	描述
triangleNum	long	三角形个数
triangleIndexNoSets	Array<long>	三角形顶点序列集合

7.3.4.2. 纹理信息

纹理图片文件是材质信息中引用的图像文件，为三维模型提供详细的表面细节和外观。纹理图片可以是颜色贴图、法线贴图、高度贴图等，与几何要素文件中的材质信息相关联，通过纹理映射技术到模型的表面上。

7.3.4.3. 属性记录文件

具体下“属性文件”。

7.4 属性文件

属性文件用于描述全空间三维模型数据相关联的属性信息，包括数据的属性结构、属性字段的元数据以及具体的属性记录值。这些属性可能包括分类信息（如建筑物、道路、植被等）、高度信息、唯一标识符等。

M3D属性文件是面向多图层的，可以高效地管理和存储多个具有不同属性的元素。通过不同的图层，可以将具有相似功能或属性的对象组合在一起。其属性文件包括图层属性信息描述文件Layerinfo.json和属性数据文件*.att。

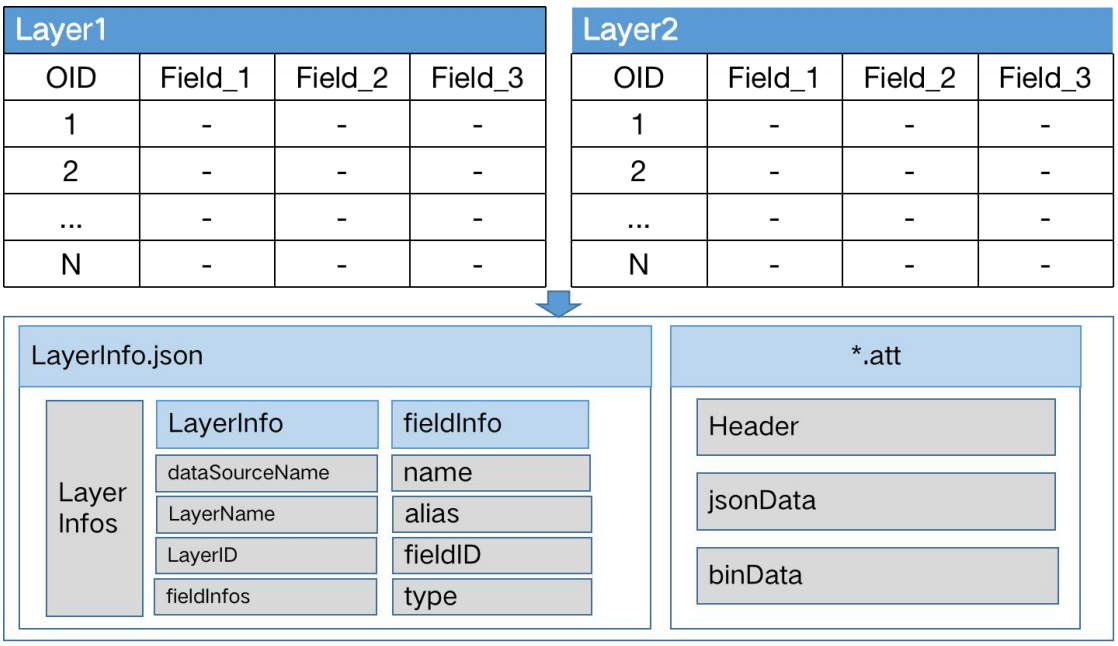


图 13　属性文件的关联关系

纹理图片文件对象说明见下表所示：

表 26　纹理图片文件对象说明

标签名	类型	描述
texType	string	纹理图片类型，取值范围｛“jpg”，“png”，“webp”，“ktx”｝
texture	Uri	纹理图片数据路径

7.4.1. 图层属性信息描述文件

属性信息描述文件是layerinfo.json文件，描述全局各图层对象的字段信息，以utf-8编码。

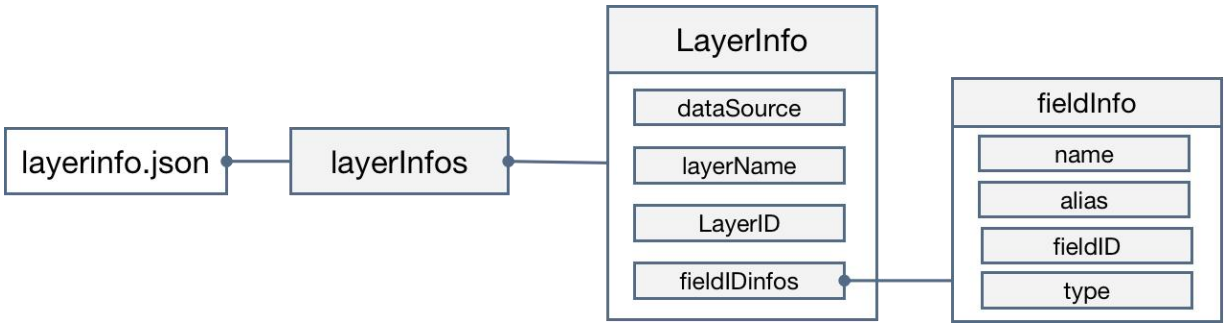


图 14　layerinfo 文件类图

Layerinfo属性描述信息文件各标签含义：

表 27　layerInfos 对象说明

标签名	类型	描述
layerIofos	Array<layerInfo>	各图层的属性描述信息集合

表 28　layerInfo 对象各标签含义

标签名	类型	描述
dataSource	string	各图层的属性描述信息集合
layerName	string	层名
layerID	uint32	层唯一标识
fieldInfos	Array<fieldInfo>	某一图层字段信息集合，用 fieldInfo 对象构成的数组表示

表 29　fieldInfo 对象各标签含义

标签名	类型	描述
name	string	字段名
alias	string	字段别名
fieldID	uint32	属性字段唯一标识
type	string	字段类型 取值范围：{‘bool’，‘byte’，‘int16’，‘uint16’，‘int32’，‘uint32’，‘int64’，‘uint64’，‘float’，‘double’，‘text', 'datetime'}

7.4.2. 属性数据文件

属性数据文件包含各图层中的属性描述信息和每个对象的各属性值，采用*.att文件存储，由attheader、jsonChunkheader+jsonChunk、binChunkheader+binChunk三部分组成。

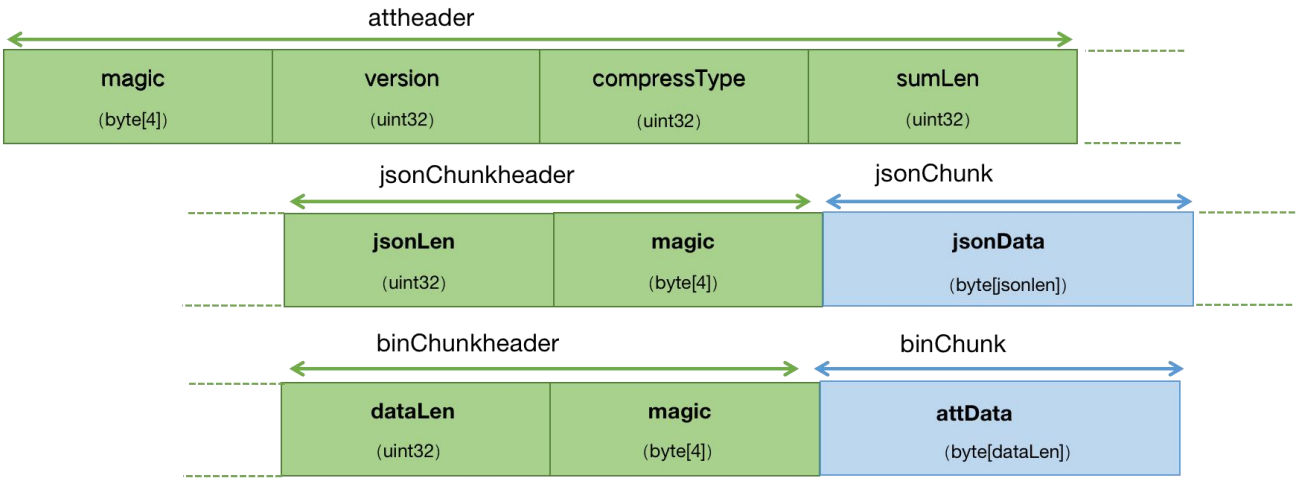


图 15　属性文件组成

表 30　属性文件对象各标签含义

组成部分	标签名	类型	描述
attheader	magic	byte[4]	文件类型，固定取值“att\0”
	version	uint32	版本，值为 1
	compressType	uint32	是否压缩：0 不压缩；1 gzip 压缩
	sumLen	uint32	数据总长
jsonChunkHeader	jsonLen	uint32	json 内容长度
	magic	byte[4]	片段类型，固定取值“json”
jsonChunk	jsonData	byte[jsonLen]	json 内容，描述数据层信息
binChunkHeader	dataLen	uint32	属性数据长度
	magic	byte[4]	片段类型，固定取值“bin\0”
binChunk	attData	byte[dataLen]	属性数据

7.4.2.1. attheader 描述

json以utf-8编码，内容组织如下图：

表 31 attheader 对象各标签含义

组成部分	标签名	类型	描述
attheader	magic	byte[4]	文件类型，固定取值“att\0”
	version	uint32	版本，值为 1
	compressType	uint32	是否压缩：0 不压缩；1 gzip 压缩
	sumLen	uint32	数据总长

7.4.2.2. jsonChunk 描述

json以utf-8编码，内容组织如下图：

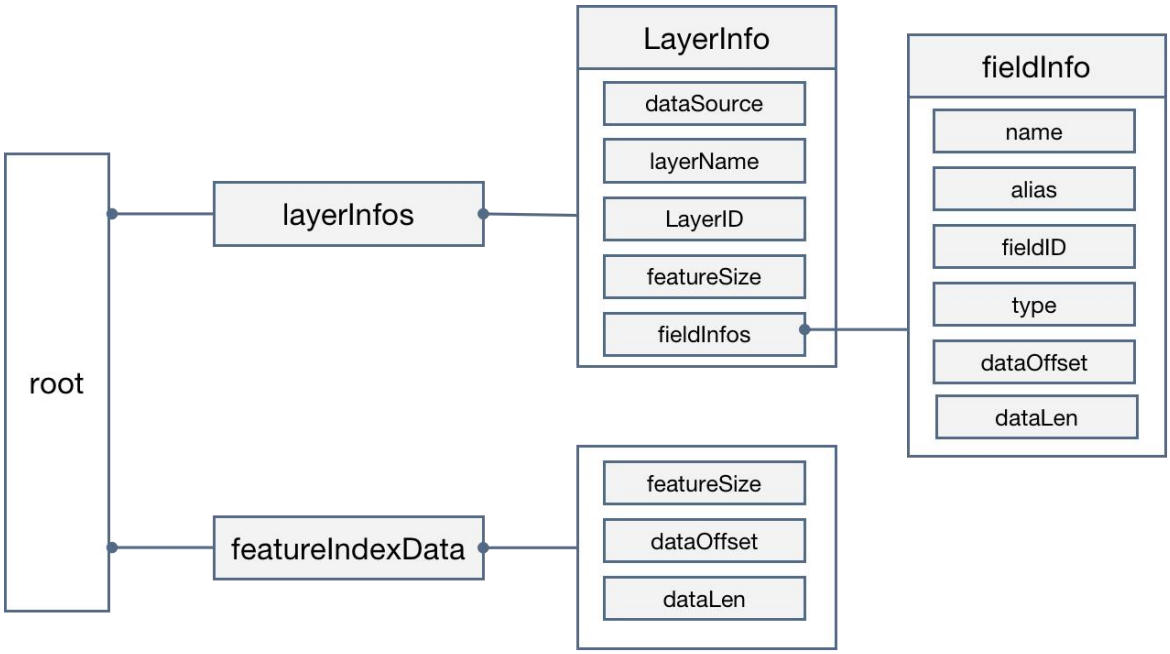


图 16　jsonChunk 组成

表 32　layerInfo 对象各标签含义

标签名	类型	描述
dataSource	string	各图层的属性描述信息集合
layerName	string	层名
layerID	uint32	层唯一标识
FeatureSize	uint32	要素个数
fieldInfos	Array<fieldInfo>	某一图层字段信息集合，用 fieldInfo 对象构成的数组表示

表 33　fieldInfo 对象各标签含义

标签名	类型	描述
name	string	字段名
alias	string	字段别名
fieldID	uint32	属性字段唯一标识
type	string	字段类型 取值范围：{‘bool’，‘byte’，‘int16’，‘uint16’，‘int32’，‘uint32’，‘int64’，‘uint64’，‘float’，‘double’，‘text', 'datetime'}
dataOffset	uint32	相对于 attData 起始位置偏移
dataLen	uint32	所有要素对应当前属性数据长度

表 34　featureIndexData 对象各标签含义

标签名	类型	描述

featureSize	uint32	要素个数
dataOffset	uint32	起始位置
dataLen	uint32	数据长度

7.4.2.3. binChunk 数据描述

1.要求

（1）对齐要求

jsonLen为8的整数倍，若不足，用0填补
att属性值起始位置均为8的整数倍，若不足，用0填补

（2）大小端

对于数字类型值均采用小端存储

（3）压缩

取值： 0表示不压缩；1表示gzip方式压缩
压缩内容：除attheader的其他部分，即jsonChunkHeader、jsonChunk、binChunkHeader、binChunk。

2.featureIndexData 数据结构

例：TID:1,layerIndex:2,featureIndex:5 存储结果为1 2 5 \0 \0 \0 \0

表 35 featureIndexData 数据结构

标签名	类型	描述
TID	uint32	全局 Index，从 0 开始
layerIndex	uint32	该瓦片下 layer 索引
featureIndex	uint32	要素索引

3.字段类型及示例

（1）bool类型数据存储

例：true、true、false存储结果为
1 1 0 \0 \0 \0 \0 \0

表 36 bool 类型数据结构

标签名	类型	描述
value	byte[featureSize]	bool 类型值序列

（2）byte类型数据存储

例：5、7、11存储结果为5 7 11 \0 \0 \0 \0 \0

表 34 byte 类型数据结构

标签名	类型	描述

value	byte[featureSize]	byte 类型值序列

（3）int16类型数据存储

例：51、74、118存储结果为51 74 118 \0 \0

表 38　int16 类型数据结构

标签名	类型	描述
value	int16[featureSize]	int16 类型值序列

（4）int32类型数据存储

例：51、74、118存储结果为51 74 118 \0 \0 \0 \0

表 39　int32 类型数据结构

标签名	类型	描述
value	int32[featureSize]	int32 类型值序列

（5）int64类型数据存储

例：51、74、118存储结果为51 74 118

表 40　int64 类型数据结构

标签名	类型	描述
value	int64[featureSize]	int64 类型值序列

（6）float类型数据存储

例：5.2、74.1、1.18存储结果为5.2 74.1 1.18 \0 \0 \0 \0

表 41　float 类型数据结构

标签名	类型	描述
value	float[featureSize]	float 类型值序列

（7）double类型数据存储

例：5.2、74.1、1.18存储结果为5.2 74.1 1.18

表 42　double 类型数据结构

标签名	类型	描述
value	double[featureSize]	double 类型值序列

（8）text类型数据存储

例：“Zondy”、“中地”、“"、null 存储结果为6 7 1 0 Zondy\0 中地\0 \0 \0 \0

表 43　text 类型数据结构

标签名	类型	描述
textSize	uint32[featureSize]	各 text 值长度序列
value	char[featureSize]	text 类型值序列，以 utf-8 编码

（9）datatime类型数据存储

按照纪元时间方式存储，即1970年1月1日00:00UTC以来所经历的毫秒数，例：北京时间2021-5-18 21:07:32存储结果为1621372052000

表 44　datatime 类型数据结构

标签名	类型	描述
value	int64[featureSize]	int64 类型值序列

（10）uint16类型数据存储

例：51、74、118存储结果为51 74 118 \0 \0

表 45　uint16 类型数据结构

标签名	类型	描述
value	uint16[featureSize]	uint16 类型值序列

（11）uint32类型数据存储

例：51、74、118存储结果为51 74 118 \0 \0 \0 \0

表 46　uint32 类型数据结构

标签名	类型	描述
value	uint32[featureSize]	uint32 类型值序列

（12）uint64 类型数据存储

例：51、74、118存储结果为51 74 118

表 47　uint64 类型数据结构

标签名	类型	描述
value	uint64[featureSize]	uint64 类型值序列

（13）各类型数据对null值支持情况

除了text类型可支持外，其余均不支持，对于null值存储为0。

7.5 结构树文件

结构树是组织和管理三维场景中所有构件的层次结构。结构树的根节点代表整个场景，每个子节点代表一个构件。子节点可以包含更多的子节点，从而形成一个完整的层次结构。通过结构树，用户可以方便地浏览和查询场景中的构件信息，也可以实现对构件的选择、高亮、隐藏等操作。

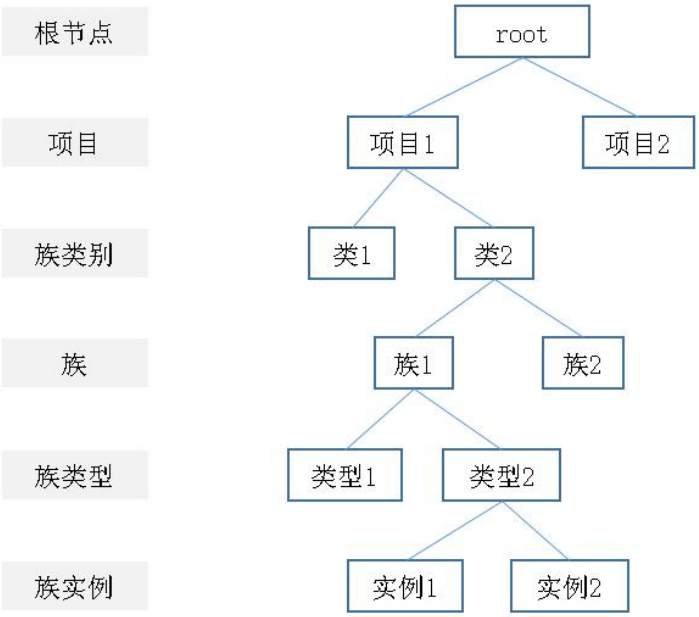


图 17 结构树组成概念图

根节点：

根节点是BIM模型或项目的最顶层结构，它代表了整个BIM模型或项目。例如，在一个大型住宅小区或工业园区的BIM模型中，整个住宅小区或工业园区就是根节点。

项目：

项目是根节点下的子节点，它代表了一个具体的建筑或基础设施项目。在大型住宅小区中，每个单独的建筑楼栋都可以被视为一个项目。这些项目可以有具体的名称或编号，以便在BIM模型中轻松识别和管理。例如，“项目1”可能代表1号楼，“项目2”代表2号楼。

族类别：

族类别是BIM中用于组织具有相似特性和用途的族的分类。这些类别有助于用户更容易地找到和管理相关的族。例如，门、窗、墙、家具等都可以是不同的族类别。族类别提供了对族进行分组和分类的方法，使得在大型BIM模型中可以高效地管理和检索族。

族：

族是BIM中的关键组成部分，它代表了一组具有相同参数化属性和行为的可重复使用的对象。族允许创建和存储具有特定属性和行为的对象，以便在多个项目或模型中使用。例如，一个门族可能包含多种不同尺寸、样式和功能的门类型。族是BIM模型中的基础构建块，可以被参数化，以便根据项目的具体需求进行调整和修改。

族类型：

族类型是族内的具体变体，它定义族的特定尺寸、形状、材料等属性。每个族可以有多个类型，以适应不同的设计需求。例如，一个门族可能包括单扇门、双扇门、滑动门等不同的类型，每种类型都有其特定的尺寸和配置。族类型允许用户根据项目需要选择合适的族变体，并对其进行定制。

族实例：

族实例是族类型在BIM模型中的具体实现，它是模型中的单个对象。在BIM模型中，每一扇门、每一扇窗户、每一堵墙等都是族实例。这些实例具有唯一的标识和位置，并可能包含特定于该实例的属性信息，如尺寸、颜色、材质等。族实例是BIM模型中的实际元素，它们构成了模型的物理和逻辑结构。例如：1 号楼的单扇门（属于门族 - 单扇门类型），是族实例，含唯一标识、位置及尺寸等属性。以分层分户的数据为例，其结构如下图所示：

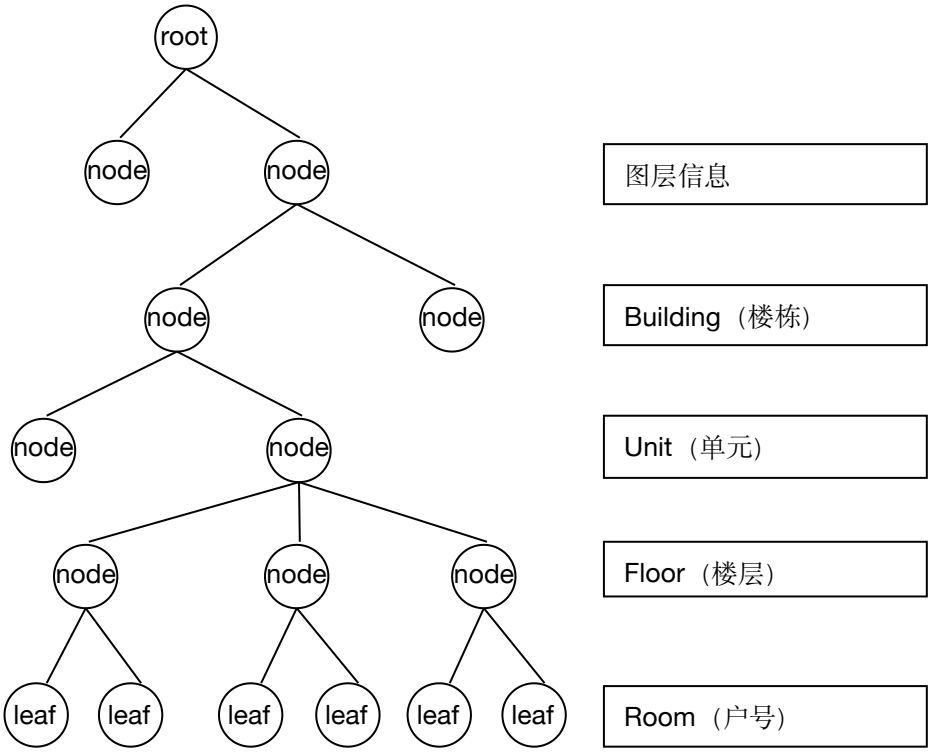


图 18　分层分户结构树组成示例

第一级节点：对应分层分户数据的图层信息
第二级节点：对应分层分户数据的楼栋信息
第三级节点：对应分层分户的单元信息
第四级节点：对应分层分户的楼层信息
第五级节点：对应分层分户的户号信息

7.5.1. 结构树描述信息文件

通过structuretree.json文件描述结构树信息：

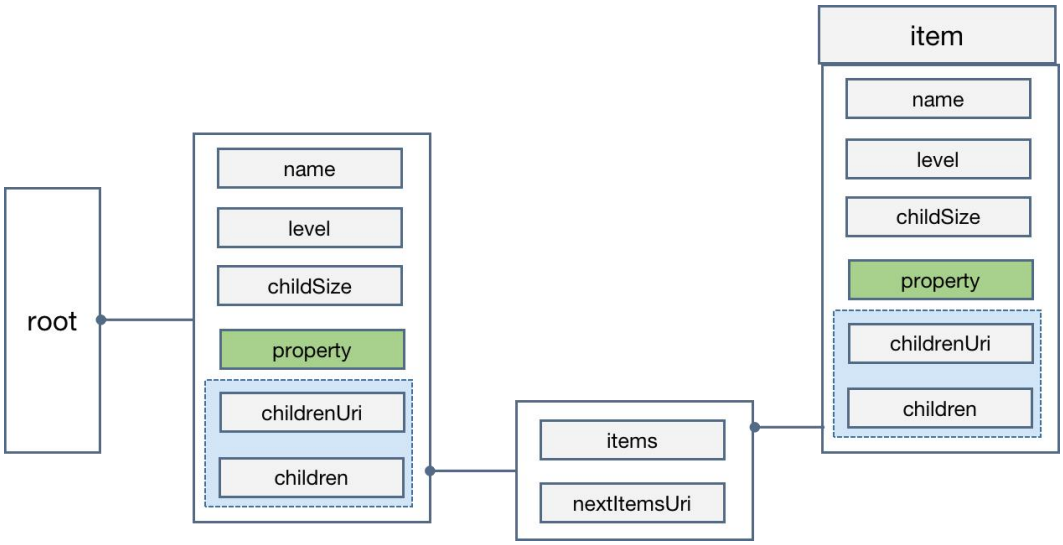


图 19　结构树对象组成类图

结构树的各对象标签信息如下表所示：

表 48　结构树 root 对象说明

标签名	类型	描述
name	string	名称
level	uint32	树深度
childSize	uint32	子节点个数
property	obj	用于记录拓展信息
childrenUri	string	子节点文件路径
children	object	子节点信息（childrenUri、children 只存在一个）

表 49　结构树 children 对象说明

标签名	类型	描述
items	array<item>	子信息序列
nextItemsUri	string	下一批 items 文件路径

表 50　结构树 item 对象说明

标签名	类型	描述
name	string	名称
level	uint32	树深度
childSize	uint32	子节点个数
property	obj	用于拓展 item 信息，可不存在
childrenUri	string	子节点文件路径
children	object	子节点信息（childrenUri、children 只存在一个）

7.5.2. 结构树节点属性描述信息文件

1.叶子节点

结构树节点属性信息描述对象是property，用于描述结构树的扩展信息，其组成为：

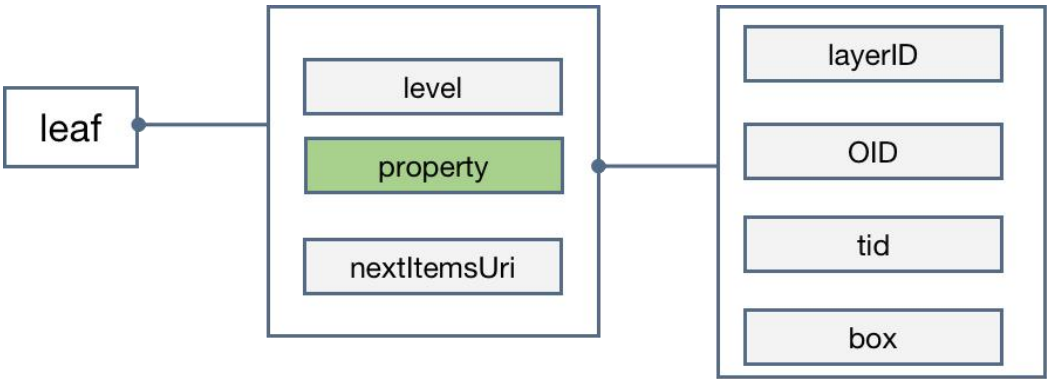


图 20　结构树扩展文件类图

描述结构树的扩展对象property叶子节点扩展对象说明如下表所示：

表 51 叶子节点 property 对象说明

标签名	类型	描述
layerID	uint32	数据层 ID
OID	uint32	要素 OID
tid	Int64	要素全局编码，结构树以深度优先的方式进行编码
box	array<double>	要素包围盒

表 52　box 对象说明

索引	类型	描述
0	double	x 最小值
1	double	y 最小值
2	double	z 最小值
3	double	x 最大值
4	double	y 最大值
5	double	z 最大值

2.非叶子节点

表 53　非叶子节点 property 对象说明

标签名	类型	描述
minTid	uint32	要素编码最小 Tid

maxlid	uint32	要素编码最大 Tid

7.5.3. 结构树节点拆分原则

为了保证每一个数据的大小，在保存文件的时候，会根据给定的限定值（当前为200），将结构树拆分为子树和兄弟树进行存储。

子树拆分规则：

统计每一级节点数的总和，当节点数大于给定的限定值时，则将该集设置为子树。

比如：有一组数据　第一级节点数1个，第二级节点数20个，第三级节点数130个，第四级节点数180个。 因为 1+20+130+180>200，则从第四级创建子树。

子树命名规则：

根据节点级别与所在级别的索引号命名。

如：0_0_0_2_7_1.json


```
"childSize": 21,
"children": {
    "items": [{
            "childSize": 15,
            "childrenUri": "structuretree/0_0_0_0_0.json",
            "level": 0,
            "name": "WQ1",
            "property": {
                "maxBatchID": 14,
                "minBatchID": 0
            }
        }, {
            "childSize": 1,
            "childrenUri": "structuretree/0_0_0_0_1.json",
            "level": 0,
            "name": "WQ2",
            "property": {
                "maxBatchID": 15,
                "minBatchID": 15
            }
        }, {
```


图 21　使用 childrenUri 指向子树的相对路径

兄弟树拆分规则：

当该级别的兄弟节点总数，大于给定的限定值时，则拆分出兄弟节点。使用 nextItemsUri 记录下一个兄弟节点的相对路径

兄弟树命名规则：

树名+page+序号。json

如：0_0_0_0_19page1.json


```
{
    "items": [...]
    "nextItemsUri": "0_0_0_0_19page1.json"
}
```

图 22　兄弟树拆分示例

7.6 Shared 公共文件

公共文件Shared包括数据中公共的几何要素文件、属性记录文件、纹理图片文件，可极大地减少存储的数据量，并支持实例化渲染。Shared文件夹对象说明见下表所示。

表 54　Shared 文件夹对象说明

标签名	类型	描述
shared	Uri	公共数据路径

8　全空间三维模型数据服务接口

8.1 概述

M3D服务接口遵循RESTful设计规范，可通过三维客户端调用该服务。

必选接口应包含以下内容：

——M3DData

——M3DSharedResources

——M3DRootNodeInfo

——M3DNodeInfo

——M3DNodeData

8.2 M3D 数据信息获取服务

数据信息（M3DData）获取服务，用于描述M3D数据的基本信息，应包括数据所有者、版本号、数据名称、数据类型、空间参考系等信息。该接口应关联获取公共资源获取、根节点信息获取两个子接口。M3DData接口说明见表37。

表 55　M3DData 接口说明

接口内容	接口内容说明
接口描述	用于获取服务的数据元数据信息，描述数据基本信息、版本号、数据名称、数据类型、空间参照系等信息
URL 模板	<base-url>services/[{folder}]<service-name>M3dServer
URL 示例	http://igserver.mapgis.com/igs/rest/services/wuhan-3d/M3dServer
方法	GET
返回值	格式为 json，content-Type 为 application/json，内容支持 gzip 编码，具体返回内容参见附录 B.1

8.3 M3D 公共资源获取服务

数据公共资源（M3DSharedResources）获取服务，应包括数据中可共享使用的材质、纹理、要素信息，如表所示。

表 56　M3DSharedResources 接口说明

接口内容	接口内容说明

接口描述	用于获取 M3D 数据下的公共资源数据，包括材质、纹理、要素信息，返回 m3d 压缩包格式的二进制数据流
URL 模板	<m3d-data-url>shared-resources
URL 示例	http://igserver.mapgis.com/igs/rest/services/wuhan-3d/M3dServer/shared-resources
方法	GET
返回值	格式为二进制数据流，content-Type 为 application/octet-stream，内容支持 gzip 编码

8.4 M3D 根节点信息获取服务

数据根节点信息（M3DRootNodeInfo）获取服务，用于描述M3D树形结构的根节点，应包括节点名称、节点LOD级别、节点外包球范围等信息，如下表所示。

表 57　M3DRootNodeInfo 接口说明

接口内容	接口内容说明
接口描述	用于获取 M3D 数据下根节点的元数据信息，描述节点名称、节点 LOD 级别、节点外包球范围等信息
URL 模板	<m3d-data-url>nodes/root
URL 示例	http://igserver.mapgis.com/igs/rest/services/wuhan-3d/M3dServer/nodes/root
方法	GET
返回值	格式为 json，content-Type 为 application/json，内容支持 gzip 编码，具体返回内容参见附录 B.2

8.5 M3D 节点描述信息获取服务

数据节点描述信息（M3DNodeInfo）获取服务，用于描述M3D树形结构的非根节点，应包括节点名称、节点LOD级别、节点外包球范围等信息。其中，叶子节点应包含数据URL资源，可获取要素、几何、属性、纹理图片数据资源，接口说明如下表所示。

表 58　M3DNodeInfo 接口说明

接口内容	接口内容说明
接口描述	用于获取数据下根节点下子节点的元数据信息，描述节点名称、节点 LOD 级别、节点外包球范围等信息
URL 模板	<m3d-data-url>nodes/<node-id>
URL 示例	http://igserver.mapgis.com/igs/rest/services/wuhan-3d/M3dServer/nodes/1
方法	GET
返回值	格式为 json，content-Type 为 application/json，内容支持 gzip 编码，具体返回内容参见附录 B.3

8.6 M3D 节点数据信息获取服务

M3D节点数据信息（M3DNodeData）获取服务，可返回节点下指定数据的二进制压缩文件，应包括几何数据信息、属性数据信息、纹理数据信息。M3D节点数据信息获取接口如下表所示。

表 59　M3DNodeData 接口说明

接口内容	接口内容说明
接口描述	用于获取与节点关联的数据信息，包括几何、属性、材质和纹理数据
URL 模板	<m3d-node-url>data/<data-name>
URL 示例	http://igserver.mapgis.com/igs/rest/services/wuhan-3d/M3dServer/nodes/1/data/1.m3d http://igserver.mapgis.com/igs/rest/services/wuhan-3d/M3dServer/nodes/1/data/1.att
方法	GET

返回值	格式为二进制数据流，content-Type 为 application/octet-stream，内容支持 gzip 编码

附录 A
（资料性）
数据示例

A.1 数据信息描述文件示例

```
{
    "asset" : "Zondy Inc.",              // string，数据基本信息，如数据所有者
    "boundingVolume" : {                          // 数据地理范围
        "boundingBox" : {
            "bottom": 0.6981191595977578,
            "left": 2.02453695048584,
            "maxHeight": 35.014613427221775,
            "minHeight": -1.2957081887871027,
            "right": 2.0246334546574776,
            "top": 0.6981737860659485
        }
    },
    "compressType" : "zip",        // 文件压缩类型
    "dataName" : "ZondyModels",       // string，数据名
    "guid" : "E1201D0B181B4F19BD51F6487DDF7F2F",     // 数据唯一标志符
    "lodType" : "REPLACE",              // LOD 类型，Add 和 Replace
    "position" : {                 // 定位点信息，数据类型为 Point
        "x" : 114.33,
        "y" : 30.5,
        "z" : 20.5
    },
    ],
    "rootNode" : {       // 根节点文件路径
        "uri" : "rootNode.json"
    }
    "spatialReference" : "wgs84",     // 空间坐标参考系，wkt、wkid 格式
    "treeType" : "QuadTree",           // 树形组织结构类型，取值 QuadTreelOCTreelK-DTreelRTree
    "version" : "2・1",                    // string，版本
}
```

A.2 （根）节点信息描述文件示例

```
{
  "boundingVolume": {        // 数据地理范围
    "boundingBox": {
      "bottom": 0.5972090831473278,
      "left": 1.901667995888956,
      "maxHeight": 486.1398727437481,
      "minHeight": 403.9074118351564,
```

```
        "right": 1.901732647625562,
        "top": 0.5972813249068002
      }
    },
    "childrenNode": [
      {
        "boundingVolume": {
          "boundingBox": {
            "bottom": 0.5972095259506565,
            "left": 1.9016679965717072,
            "maxHeight": 445.4516488024965,
            "minHeight": 403.9012103090063,
            "right": 1.9017061884005992,
            "top": 0.5972501940129736
          }
        },
        "lodError": 24,
        "uri": ".node/0/0.json"
      },
      {
        "boundingVolume": {
          "boundingBox": {
            "bottom": 0.5972090832334465,
            "left": 1.901694238978699,
            "maxHeight": 485.17136704269797,
            "minHeight": 417.99612840265036,
            "right": 1.9017324343808204,
            "top": 0.597249977750888
          }
        },
        "lodError": 24,
        "uri": ".node/1/1.json"
      },
      {
        "boundingVolume": {
          "boundingBox": {
            "bottom": 0.5972479277284803,
            "left": 1.901668566894992,
            "maxHeight": 486.13951092679054,
            "minHeight": 418.22859179601073,
            "right": 1.9017065662334092,
```

```
                "top": 0.5972807296059183
            }
          },
          "lodError": 24,
          "uri": "./node/2/2.json"
        },
        {
          "boundingVolume": {
            "boundingBox": {
              "bottom": 0.5972478905577846,
              "left": 1.9016946400587305,
              "maxHeight": 437.5296869724989,
              "minHeight": 420.09787761606276,
              "right": 1.901732647543741,
              "top": 0.5972813250508775
            }
          },
          "lodError": 24,
          "uri": "./node/3/3.json"
        }
    ],
    "lodError": 480,          // Lod 切换误差
    "lodLevel": 0,                  // Lod 级别
    "lodType": "REPLACE",     // LOD 替换策略，Add 和 Replace
    "name": "rootNode"          // 名称
}
```

A.3 属性信息描述文件示例

```
{
    "dataSource": "",    // 数据资源地址
    "fieldInfos": [        //某一图层字段信息集合
      {
        "alias": "OID",       //字段别名
        "fieldID": 1438645453,        //属性字段唯一标识
        "name": "OID",            //字段名
        "type": "int64"       //字段类型
      },
      {
        "alias": "FeaName",
        "fieldID": 2330287610,
        "name": "FeaName",
        "type": "text"
      },
```

```
        {
          "alias": "FilePath",
          "fieldID": 2829547352,
          "name": "FilePath",
          "type": "text"
        }
      ],
      "layerID": 2550022088,        //层唯一标识

      "layerName": "01A01JZ"      //层名

}
```

A.4 结构树信息描述文件示例

```
      {
        "childSize": 1,                          //字节点个数
        "children": {                            //子节点信息
          "items": [                             //子信息序列
            {
              "childSize": 203,
              "childrenUri": "structuretree/0_0.json",          //子节点文件路径
              "level": 1,         //树深度
              "name": "01A01JZ",                //名称
              "property": {                     //拓展信息
                "maxTid": 203,                  //要素最大编码
                "minTid": 1                     //要素最小编码
              }
            }
          ]
        },
        "level": 0,
        "name": "RootNode",
```

```
    "property": {
      "maxTid": 203,
      "minTid": 1
    }
```

附录 B
（资料性）
服务示例

B.1 M3D节点数据信息获取服务示例

```
{
  "asset": "", //string,数据基本信息，如数据所有者
  "version": "", //string,版本
  "dataName": "", //string,数据名
  "guid": "",
  "compressType": "", //压缩类型 zip|7z|rar
  "spatialReference": {}, //wkt、wkid 格式
  "treeType": "", //树形组织结构类型，取值 QuadTree|OCTree|K-DTree|RTree
  "lodType": "", //LOD 类型，Add|Replace
  "boundingVolume": {}, // BoundingVolume
  "position": {}, //Point,
  "rootNode": "", //根节点资源路径
  "fieldInfo": {}, //属性结构
  "children": [
    {
      "id": "shared-resources",
      "url": "./shared-resources",
      "description": "m3d 服务的公共资源"
    },
    {
      "id": "root-node",
      "url": "./nodes/root",
      "description": "m3d 服务的索引数根节点"
    }
  ]
}
```

B.2 M3D根节点信息获取服务示例

```
{
```

```
    "name": "", //string,  节点名称
    "lodLevel": 0, //int,  节点 LOD 级别
    "boundingVolume": {
       "boundingSphere": {
          "center": {
                "x" : -2.708740234375,
                "y" : 62.51482391357422,
                "z" : 47.018324851989746
          },
          "radius": 473.84556
       }
    }, //boundingVolume,节点外包范围
    "lodMode": "distance", //string,LOD 切换模式,  取值范围{"distance","pixel"}
    "lodError": 0.0001, //number,误差
    "transForm": 0.0001, //number,节点相对转换矩阵
    "childrenNode": [], //子节点
    "shared": "", //公共数据路径
    "children": [
       {
          "id": "0",
          "url": "../0"
       },
       {
          "id": "1",
          "url": "../1"
       }
    ]
}
```

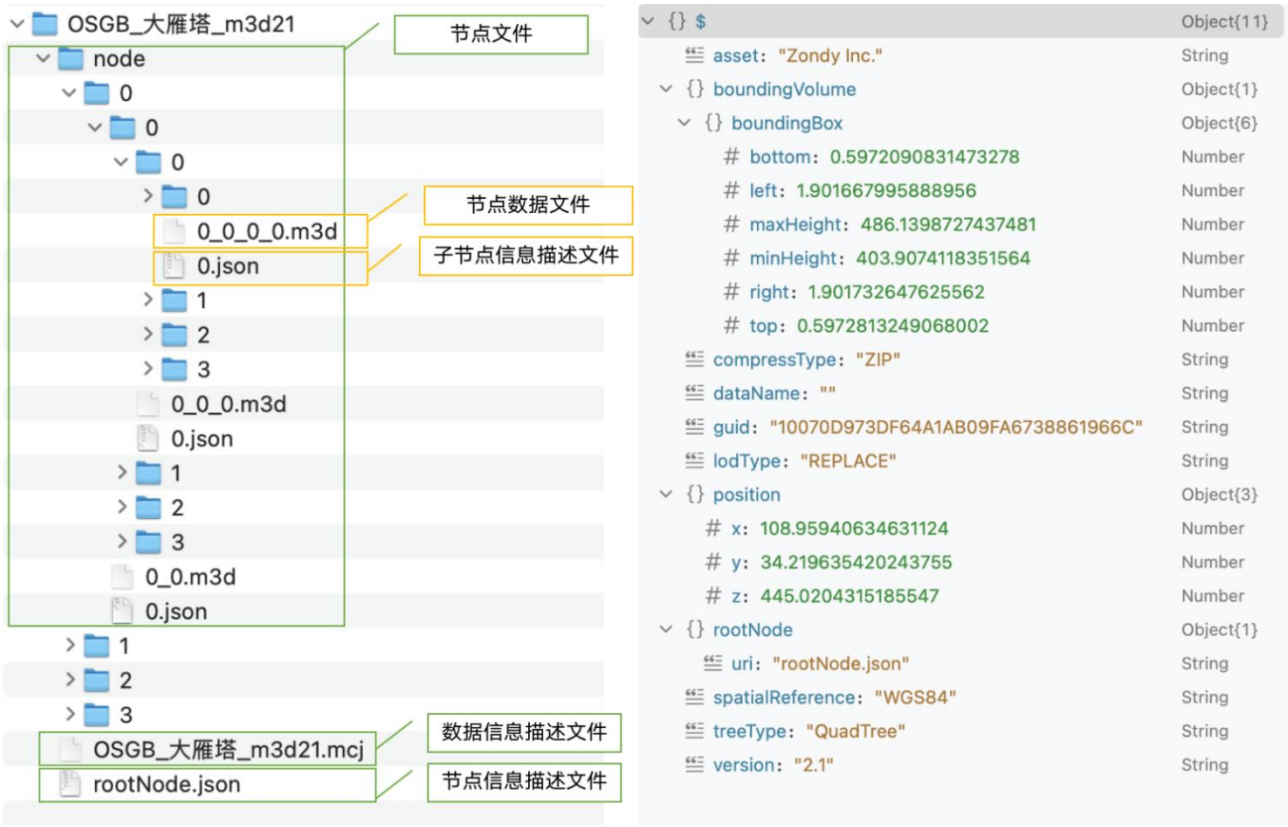
B.3 M3D节点描述信息获取服务示例

```
{
       "name": "", //string,  节点名称
       "lodLevel": 0, //int,  节点 LOD 级别
       "boundingVolume": {
             "boundingSphere": {
                   "center": {
                         "x" : -2.708740234375,
                         "y" : 62.51482391357422,
                         "z" : 47.018324851989746
                         },
```

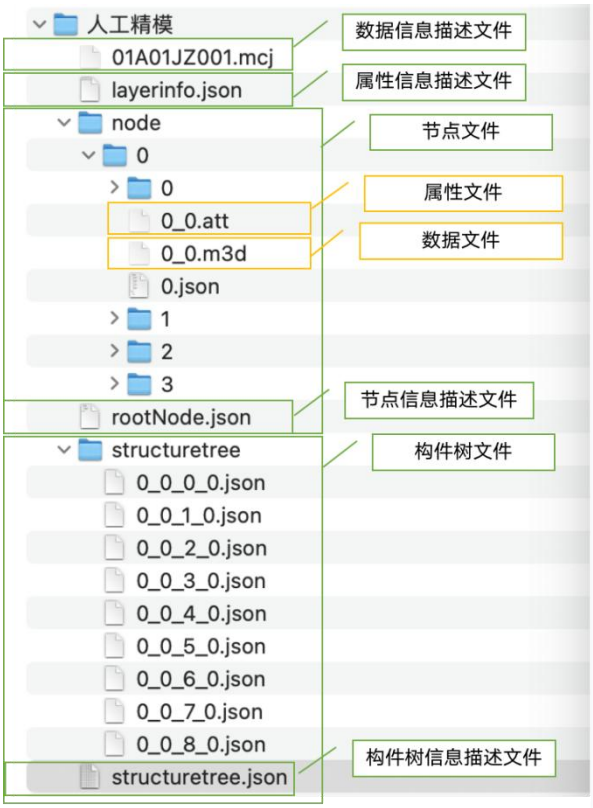
```
            "radius": 473.84556
        }
},//boundingVolume,节点外包范围
"lodMode": "distance", //string,LOD 切换模式，取值范围{"distance","pixel"}
"lodError": 0.0001, //number,误差
"transForm": 0.0001, //number,节点相对转换矩阵
"parentNode": {}, //父节点
"childrenNode": [], //子节点
"shared": "", //公共数据路径
"children": [
    {
        "id": "0",
        "url": "../0"
    },
    {
        "id": "1",
        "url": "../1"
    }
],
"resources": {
    "geometries": [
        {
            "id": "0",
            "blobType" : "Model",
            "url": "./geometries/0"
        }
    ]
    },
    "attributes": [
        {
            "id": "0",
            "attType": "json",    // 属性类型  json | bin
            "url": "./attributes/0"
        }
    ],
    "textures": [
        {
            "id": "0",
            "url": "./textures/0"
        }
    ]
```

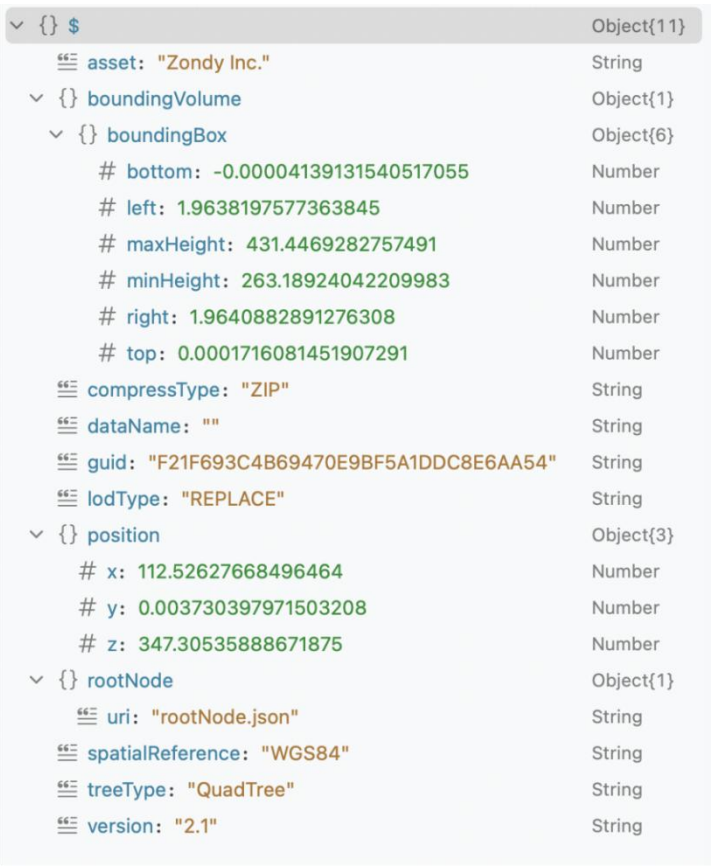
附录 C
（资料性）
不同类型数据示例

C.1 倾斜摄影数据M3D文件组成示例

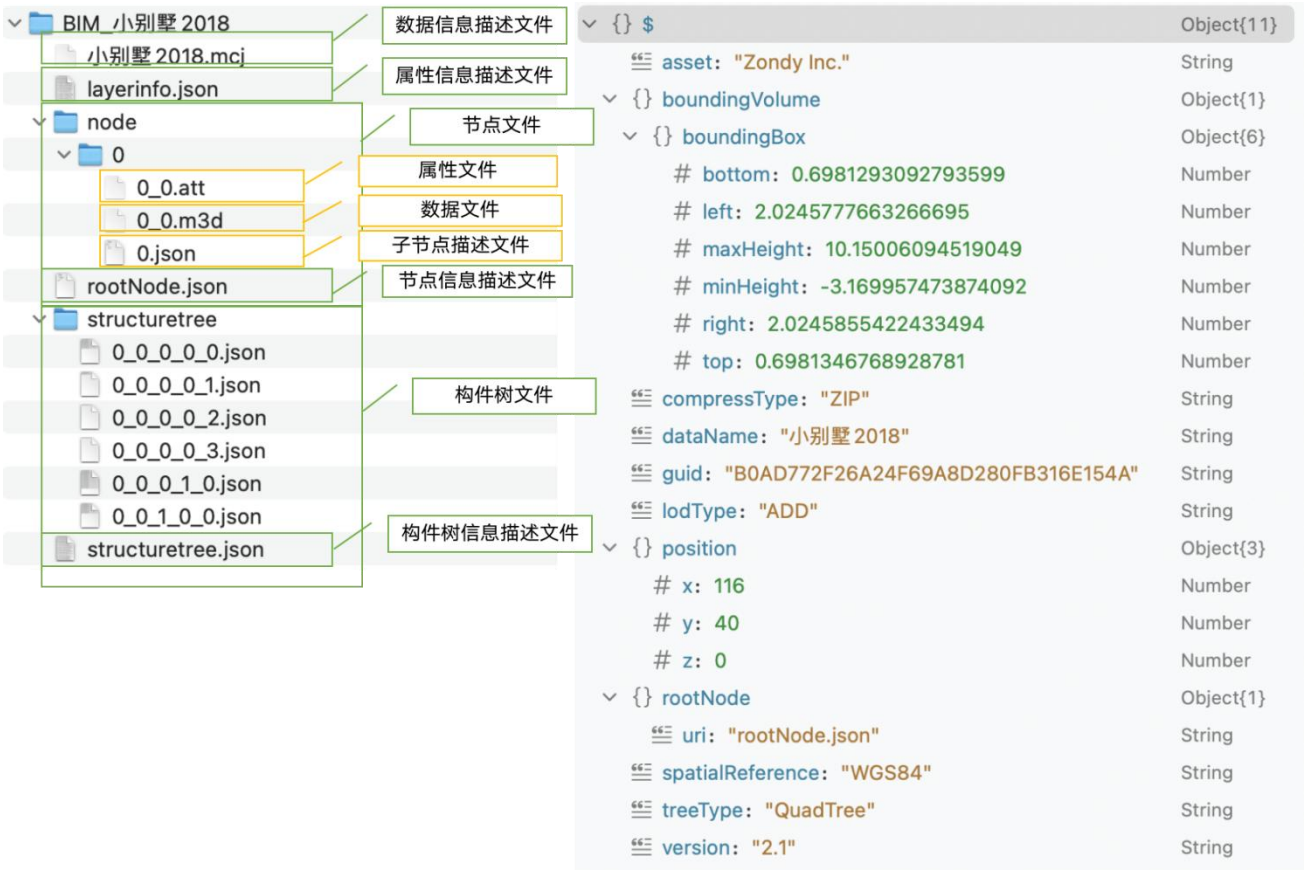


C.2 人工精模数据M3D文件组成示例

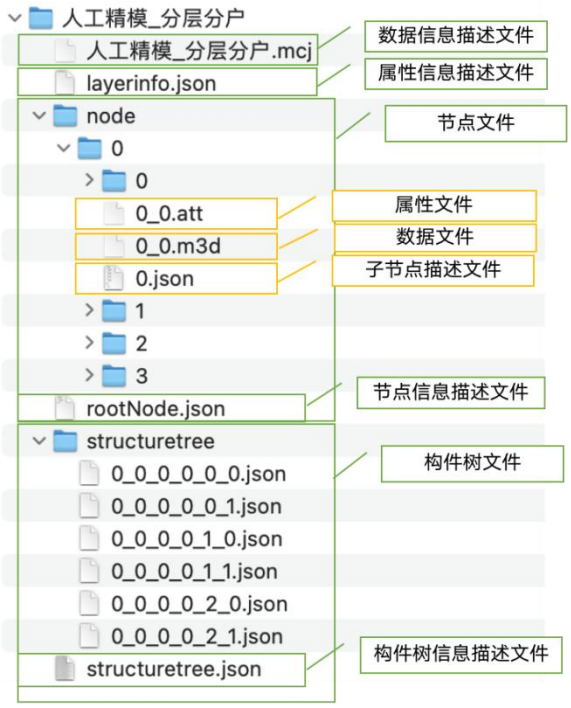


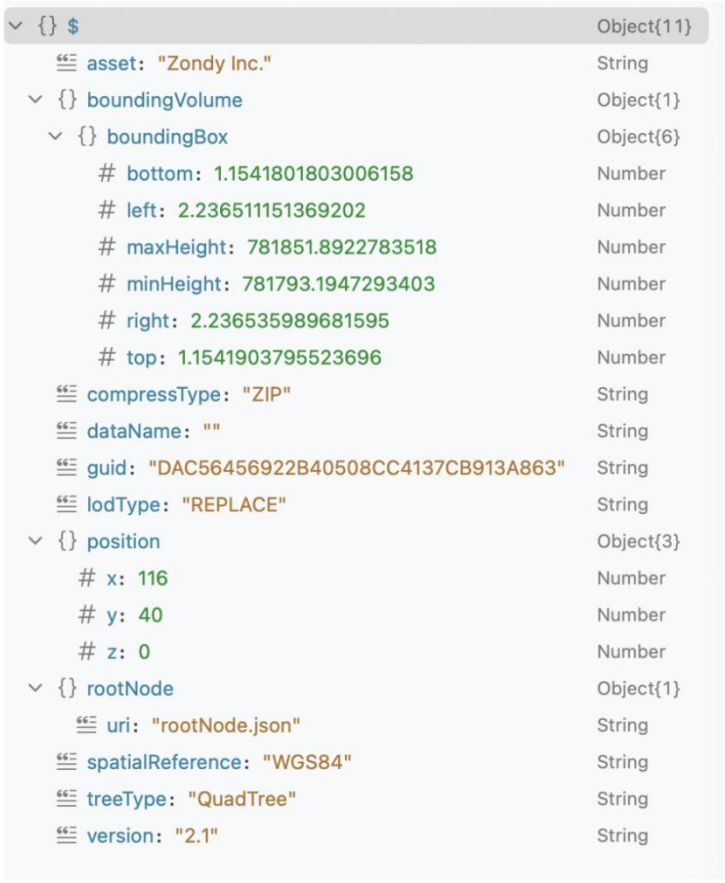


C.3 BIM数据M3D文件组成示例

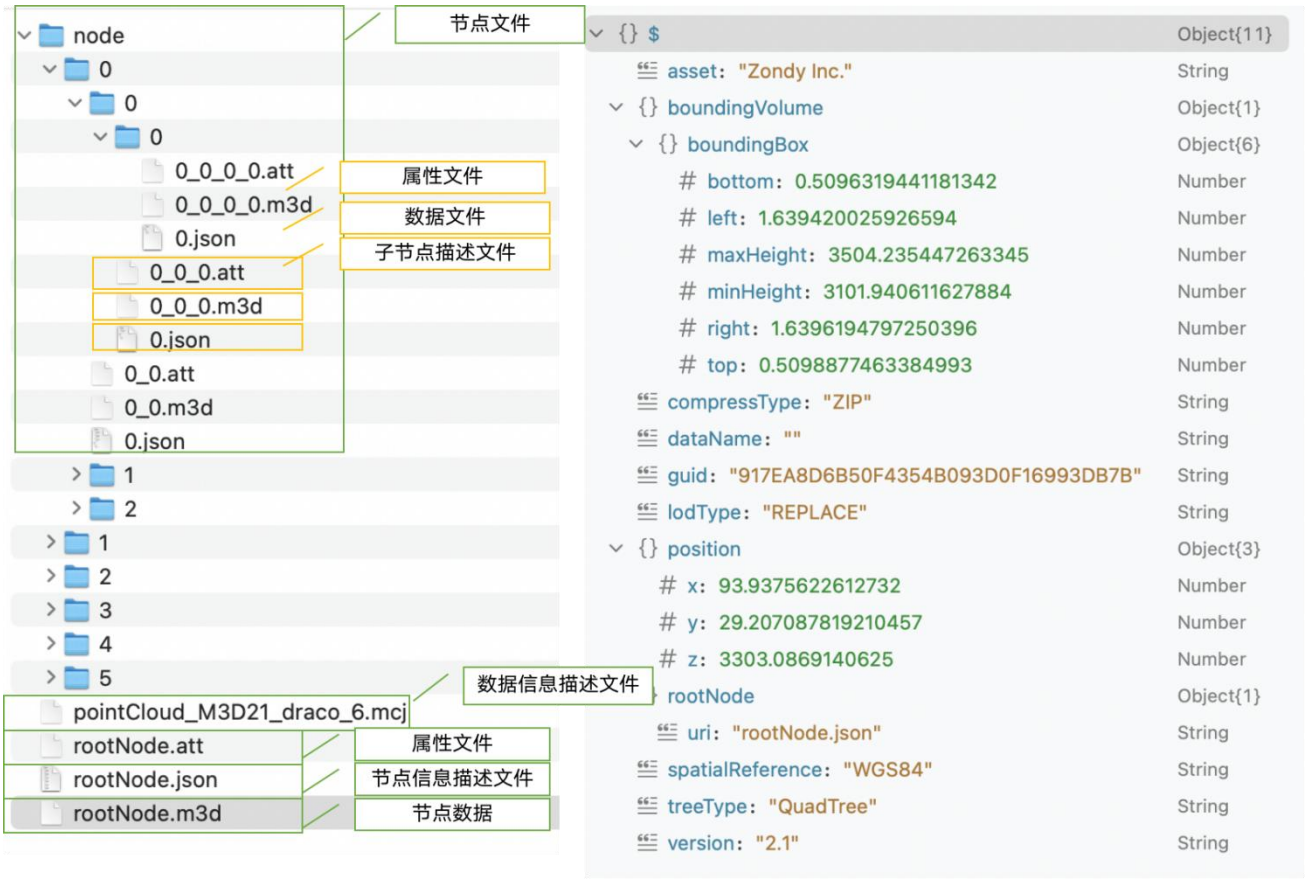


C.4 分层分户数据M3D文件组成示例





C.5 点云数据M3D文件组成示例



参考文献

[1] GB/T 23 708-2009 地理信息 地理标记语言 (GML) (ISO 19136:2007)
[2] ISO 19119:2016 Geographic information-Sercice
[3] Open Geospatial Consortium, 《OGC City Geography Markup Language(CityGML) En-coding Standard》
[4] Open Geospatial Consortium,《OGC Indexed 3d Scene Layer (I3S) and Scene Layer Package Format Specification》
[5] CityGML: http://www.opengeospatial.org/standards/citygml/
[6] glTF: https://github.com/KhronosGroup/glTF/