OPTIMIZING

Large BIM Models
with 3DTiles and glTF Technology

BHSOFT

Content

Section 1:

Introduction

1.1: What is gltf

glTF (GL Transmission Format) is an open standard file format for 3D scenes and models. It was created by the Khronos Group.

glTF is designed to be a lightweight and efficient format for transmitting 3D models over the internet. It supports a wide range of 3D data, including geometry, materials, textures, animations, and more.

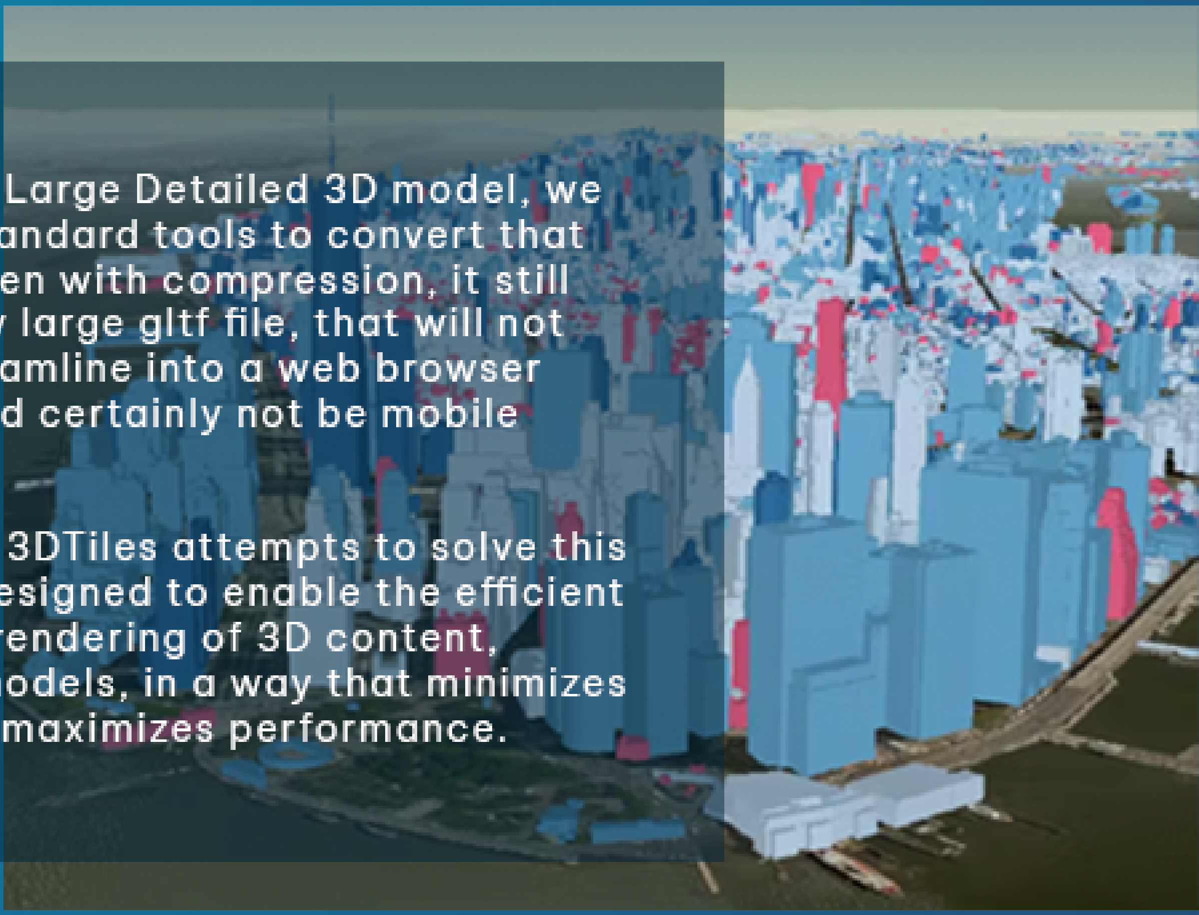
1.2: What is 3DTiles

3DTiles is a specification for streaming and rendering 3D geospatial content over the web, and it allows for the efficient transmission and display of large geospatial datasets such as Photogrammetry, 3D Buildings, BIM/CAD, Instanced Features, and Point Clouds. It defines a hierarchical data structure and a set of tile formats that deliver renderable content.

3DTiles is maintained by the Open Geospatial Consortium (OGC). The OGC is a global consortium that develops geospatial standards to facilitate interoperability and sharing of geospatial data and services.

Section 2:

How to scale gltf with 3D Tiles



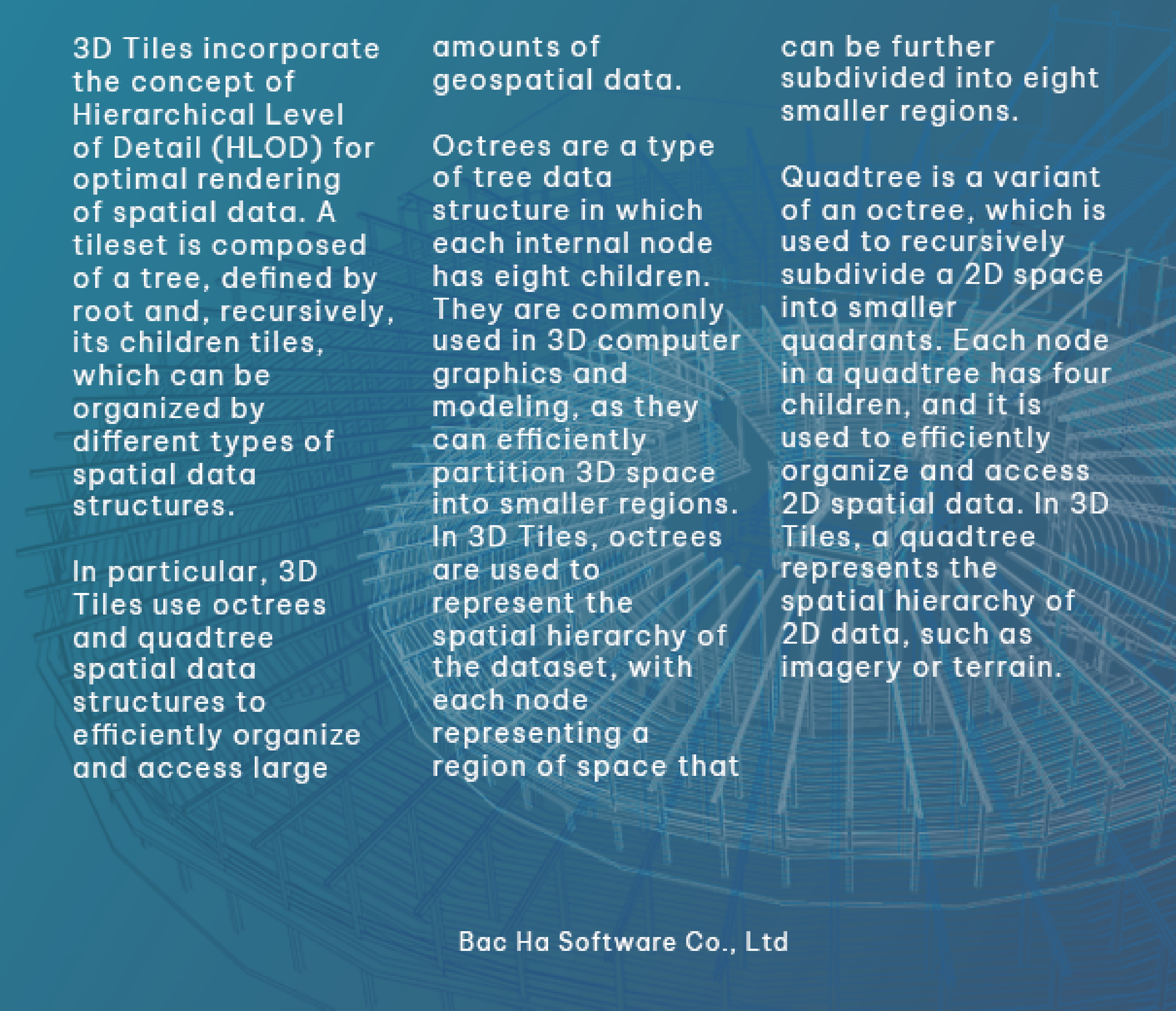
Starting with a Large Detailed 3D model, we have a lot of standard tools to convert that into gltf file, Even with compression, it still results in a very large gltf file, that will not suitable to streamline into a web browser environment and certainly not be mobile friendly.

And this is how 3DTiles attempts to solve this problem. It is designed to enable the efficient streaming and rendering of 3D content, including BIM models, in a way that minimizes load times and maximizes performance.

It's built on gltf and provides a way to express the hierarchical level of detail so you have leaves at the highest resolution and then each parent of the tree is like a simplified version of its children. And this allows you to only stream what you need for a given View.

3DTiles is essentially a JSON format specifying the bounding volume hierarchy the geometric error and the refinement type and then this creates a tree of tiles and each tile points to a gltf file. And then within each gltf we have the actual geometry and texture, compression, and metadata.

Section 2.1: Spatial data structures



3D Tiles incorporate the concept of Hierarchical Level of Detail (HLOD) for optimal rendering of spatial data. A tileset is composed of a tree, defined by root and, recursively, its children tiles, which can be organized by different types of spatial data structures.

In particular, 3D Tiles use octrees and quadtree spatial data structures to efficiently organize and access large amounts of geospatial data.

Octrees are a type of tree data structure in which each internal node has eight children. They are commonly used in 3D computer graphics and modeling, as they can efficiently partition 3D space into smaller regions. In 3D Tiles, octrees are used to represent the spatial hierarchy of the dataset, with each node representing a region of space that can be further subdivided into eight smaller regions.

Quadtree is a variant of an octree, which is used to recursively subdivide a 2D space into smaller quadrants. Each node in a quadtree has four children, and it is used to efficiently organize and access 2D spatial data. In 3D Tiles, a quadtree represents the spatial hierarchy of 2D data, such as imagery or terrain.

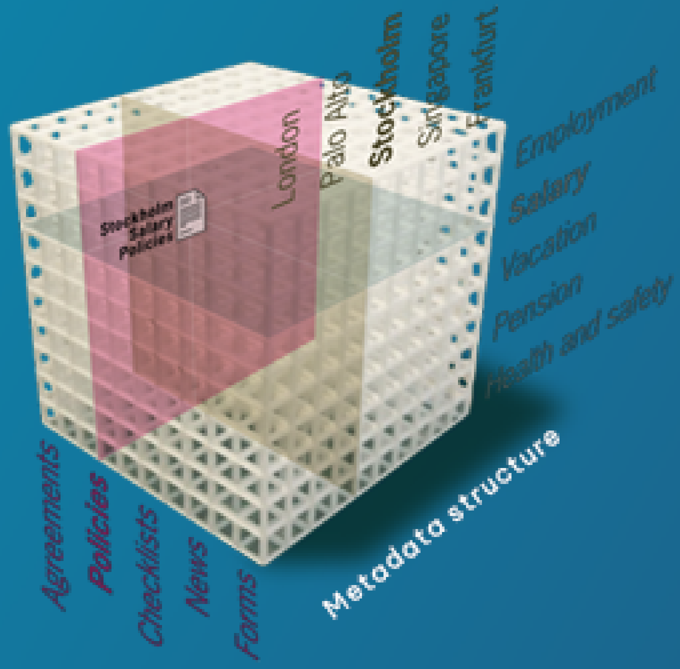
The use of these spatial data structures in 3D Tiles allows for efficient streaming and rendering of large-scale geospatial datasets. The octree and quadtree structures provide a way to break down the dataset into smaller, manageable chunks, which can be streamed to the client on-demad.

By leveraging these structures, 3D Tiles can deliver a seamless and immersive experience for users, allowing them to explore and interact with massive geospatial datasets in real time.

The 3D Tiles methodology provides a high degree of flexibility in creating various spatial data structures, which allows for the development of different 3D tiling pipelines that can generate optimized 3D tilesets for various types of input data.

For instance, a city model created using extruded footprints and a model of the same city obtained via LIDAR may require different subdivisions based on the density of the data. However, as long as the subdivisions adhere to the spatial coherence rule, where the child tiles are spatially located within the bounding volume of the parent tile, the resulting 3D tilesets can be streamed using the same 3D Tiles runtime engine. This capability is a significant factor in the 3D Tiles ecosystem, as any 3D Tiles runtime engine can stream 3D Tiles from any data source.

Section 2.2: 3D Metadata



3D metadata is a crucial component in the context of 3D tiling and streaming, as it provides additional information that can enhance the understanding and usability of 3D models. Another advantage of using 3D Tiles and glTF is the ability to easily add additional data sources to the system.

In the case of 3D Tiles, metadata is stored in the tileset.json file, which provides information about the tileset's structure, properties, and metadata. The tileset's metadata can be used to describe properties suchas the coordinate system, the spatial resolution, the units of measurement, and the source of the data. This metadata can be used to optimize the streaming of 3D Tiles, as well as to improve the processing of the data on the client side.

glTF also provides mechanisms for storing metadata, such as custom properties that can be attached to individual objects in the model. This metadata can be used to describe various properties of the 3D model, such as materials, textures, animations, and other attributes that may be relevant to the specific object.

Section 2.3: Compression

Draco is a library for compressing and decompressing 3D geometric meshes and point clouds developed by Google to compress and decompress 3D meshes. It compresses vertex positions, normals, colors, texture coordinates, and any other generic vertex attributes. The main advantage of using Draco compression is that it enhances the efficiency and speed of transmitting 3D content over the web.

gltf provides KHR_draco_mesh_ compression extension which enables loading buffers containing Draco compressed geometry. The 3D Tiles engine can load and render the model progressively, based on the user's view and level of detail. The glTF model can be decompressed and rendered using a WebGL-based renderer. With Draco, it allows for smaller file sizes and faster streaming.

Section 3:
3D Tiles Next

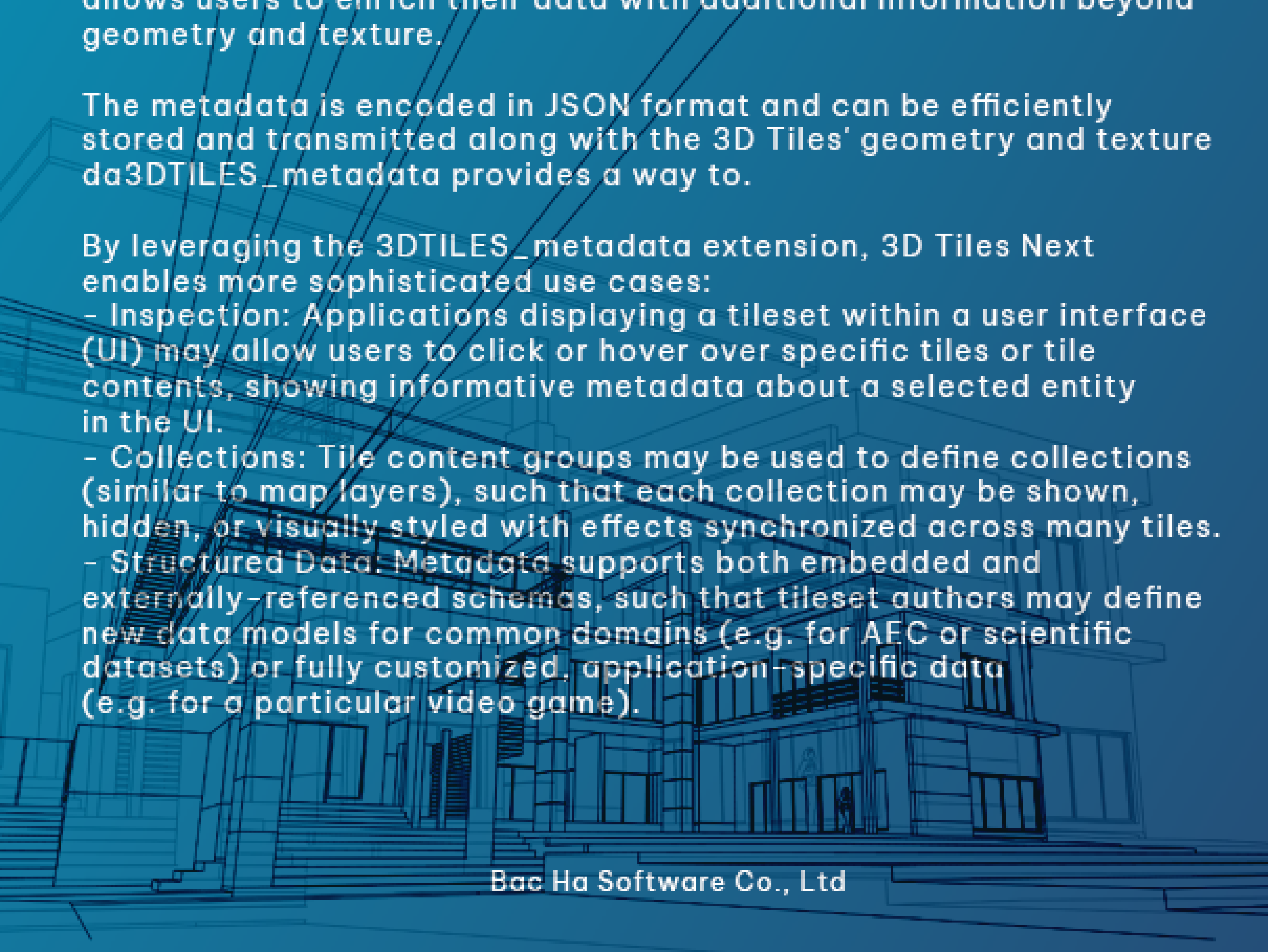
As of May 4, 2023. OGC adopts 3D Tiles Version 1.1 (aka 3D Tiles Next) 3D Tiles 1.1 promotes several 3D Tiles 1.0 extensions to 'core' and introduces new glTF™ extensions for fine-grained metadata storage.

The primary enhancements in the 3D Tiles version 1.1 include:
- 3DTILES_metadata: Semantic metadata at multiple granularities;
- 3DTILES_implicit_tiling: Implicit tiling for improved analytics and random access to tiles;
- 3DTILES_multiple_contents: Multiple contents per tile to support layering and content groupings;
- Direct references to glTF™ content for better integration with the glTF™ ecosystem

3D Tiles Next come with 3DTILES_metadata. This extension provides a way to include structured metadata ("properties") in 3D Tiles, this allows users to enrich their data with additional information beyond geometry and texture.

The metadata is encoded in JSON format and can be efficiently stored and transmitted along with the 3D Tiles' geometry and texture da3DTILES_metadata provides a way to.

By leveraging the 3DTILES_metadata extension, 3D Tiles Next enables more sophisticated use cases:
- Inspection: Applications displaying a tileset within a user interface (UI) may allow users to click or hover over specific tiles or tile contents, showing informative metadata about a selected entity in the UI.
- Collections: Tile content groups may be used to define collections (similar to map layers), such that each collection may be shown, hidden, or visually styled with effects synchronized across many tiles.
- Structured Data: Metadata supports both embedded and externally-referenced schemas, such that tileset authors may define new data models for common domains (e.g. for AEC or scientific datasets) or fully customized, application-specific data (e.g. for a particular video game).



- Optimization: Per-content metadata may include properties with performance-related semantics, enabling engines to optimize traversal and streaming algorithms significantly.

3DTILES_implicit_tiling is another new extension of 3D Tiles Next that defines a concise representation of quadtrees and octrees in 3D Tiles. This regular pattern allows for random access of tiles based on their tile coordinates which enable accelerated spatial queries, new traversal algorithms, and efficient updates of tile content, among other use cases.

The idea behind implicit tiling is to allow the creation of 3D tilesets without explicitly dividing the data into a fixed hierarchy of tiles. Instead, the data is partitioned into a set of chunks, each of which can be streamed independently.

This approach provides several benefits. his approach reduces the complexity of the tiling pipeline and simplifies the process of creating optimized 3D tilesets for large and complex datasets.

The 3DTILES_multiple_contents extension allows for multiple tilesets to be combined into a single tileset, enabling efficient streaming of complex scenes with many different models and datasets. Contents can be organized in various ways – e.g. as map layers or arbitrary groupings – which becomes particularly useful when combined with content group metadata defined by 3DTILES_ metadata.

Groups of content can be used for selectively showing content or applying custom styling. Besides styling, groups can also be used to filter out unused content resources to reduce bandwidth usage.