CDB and OpenFlight

# OpenFlight Constructs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| OpenFlight Construct | CDB Requirement/Guidelines (all from Chapter 6) | Notes | glTF Equivalent | New glTF extension required | Mapping OF to glTF difficulty | Implemented in converter prototype |
| Node Hierarchy | Req 2 – Global Zone  Req 3 – Model Zone  Chap 6.5.6.1 – Model Landing Zone  Req 17 – Model Footprint Zone  Req 18 – Model Footprint Hierarchy  Req 19 – Model Cutout Zone  Req 20 – Model Cutout Geometry  Req 21 – Model Pseudo-Interior Zone  Req 22 – Model Interior Zone  Req 4 – Layers | Chap 6.2 – Using hierarchy to organize models helps identify components of intereset | Node Hierarchy fully supported | No | Easy | Yes |
| Comment Attribute | Req 14 – Zone Names  Chap 6.5.3.1 – Zone Material Index  Req 16 – Hot Spot Temperature  Req 23 – Model Point (Damage)  Req 24 – Model Point (DIS Origin)  Req 25 – Model Point (View Point)  Chap 6.6.2.3 – Model Attach Point  Chap 6.6.2.4 – Model Anchor Point  Chap 6.6.2.5 – Model Center of Mass  Req 29 – Damage State (Levels)  Req 36 – Model Vendor Attributes | Comment field is used to extend OpenFlight node for CDB specialization for many uses  A number of constructs are available in the OF spec that could be natively encoded in the format instead of using the comment field. | “extras” property allows storage of an object with any number of named properties | No | Easy |  |
| Node Attribution | Req 13 – Model Zone Bounding Box | Group node has bounding box attribute  Other attributes available on the node:  Roof Flag  Culture Footprint | Primitives have min/max properties.  For general nodes, we could store this in the extras property | No | Easy |  |
| External Referencing | Chap 6.2.4 – Model Master File  Req 6 – Use XRef node to reference other files  Chap 6.2.5.1 – Models straddling multiple files  Chap 6.2.5.2 – Models with multiple LODs | Model Master File (comprised of xrefs) ensures a Model is seen as a single “object” even though its constituents are stored in separate files | No external referencing | Yes | Difficult | Partial  The converter can parse external references and combine them into one gltf file (using mesh instancing for repeated references) |
| Level of Detail Node | Chap 6.2.5.2 – Models with multiple LODs  Req 26 – Significant Size  Chap 6.8.1 – Exchange LODs  Chap 6.8.2 – Additive LODs | LOD node contained in XRefs in master file  Significant Size attribute on LOD  LOD node support exchange and additive LOD strategies | Requires extension (MSFT\_lod)  No sig size. Works with switch in/out | Yes | Medium | Yes  Uses MSFT\_lod extension |
| Switch Node | Req 27 – Switch Masks (one per state)  Req 28 – Switch Mask Names  Req 29 – Damage State (Transition)  Req 31 – Blur State (Transition) | Switch node supports multiple masks. Each mask can be named. | No native support for this. | Yes | Medium |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Degree of Freedom Node | Req 32 – Articulation  Req 33 – Gimbal Limits | DOF node supports min/max limits for each degree of articulation (translation, scale, rotation) | Skins, Joints, Animations.  glTF is more flexible/complex than OF when it comes to animated models | No | Medium |  |
| Light Point Node | Req 35 – Model Light Points | Light Point node can represent individual points or light ‘string’ | No native support.  Existing extensions for light sources but these are a different concept than light points | Yes | Difficult |  |
| Projection | Req 1 – Specify Projections | Required projections for GTModel, GSModel, MModel and T2DModel | Could be specified in extras property at the scene level | No | Easy |  |
| Coordinate System | Req 7 – X (left/right), Y (front/back), Z (bottom/top)  Req 8 – Origin (0,0,0) | These are native OpenFlight conventions | glTF 2.0 uses a right-handed coordinate system, with x point right, y point up and z backward  Changing this requires an extension and would reduce performance and interoperability  Would recommend keeping glTF’s axis system and adjusting the standard if needed. | No | N/A |  |
| Local Coordinate System | Chap 6.3.1.2 | Transformation Matrix is used to specify LCS | Transformations on nodes | No | Easy | Yes |
| Units | Char 6.3.1.3 | Header attribute is used to specify Units | Can be specified in extras property | No | Easy |  |
| Instancing | Req 11 – Avoid repeating identical pieces of geometry | Efficiency – smaller database size | Multiple nodes can instantiate the same mesh.  However, there is no concept of node instancing. OF is more flexible | No | Medium | Yes |
| Mesh | Req 11 – Favor mesh over polygons | Efficiency - smaller database size, fewer graphics states | Mesh is supported and highly recommended over polygons.  In OF many models use individual polygon nodes, but this would be inefficient in glTF. May lead to large geojson files. | No | Easy | Yes  Polygon nodes at the same level with the same textures are merged into one mesh |
| Vertex Ordering | Req 11 – CCW order of verts define polygon ‘front’ |  | GLTF uses CCW ordering of vertices | No | Easy | Yes |
| Relative Priority | Req 12 – Layers of coplanar geometry | Relative Priority attribute at :   * Face * Mesh * Object * Group | Not supported natively. Could be stored in “extras” | No | Easy |  |
| Textures | Req 37 – Textures stored in separate files from models  Req 41 – Relative Texture Paths  Req 42 – Object Shadow Attribute | Loading efficiency | Textures supported.  Materials in glTF are similar to extended materials in OF, but not all layers from openflight exist in glTF.  Ex: Light map, specular map, reflection map.  Material textures are not a concept in gltf. Would require extension. | Yes | Hard |  |

# Supporting more than CDB 1.X:

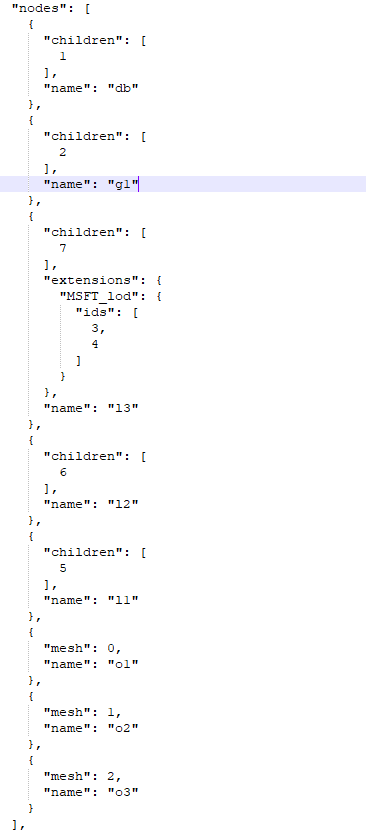
OpenFlight capabilities that could be leveraged:

* Extensions
* Extended Materials
* Hotspots
* LOD Transitions
* Cultural Footprint
* Point Nodes (Model Points instead of using tranforms)

OpenFlight to glTF converter

30 july 2020:

* Command line tool converting OpenFlight file into glTF
* Node hierarchy support and ported into glTF json file
* Polygons are converted to mesh on a per group basis
* No texture support yet
* LOD: using MSFT\_LOD extension
  + LOD attribution for Significant size is not present
  + LOD attribution ….



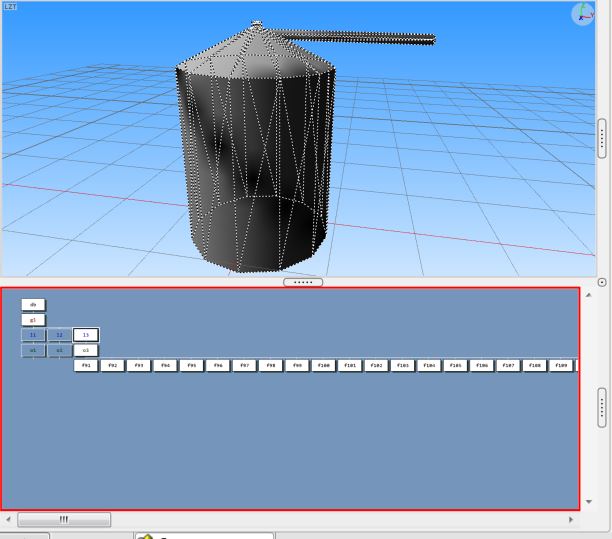
Polygons converted to mesh

Group node under LOD

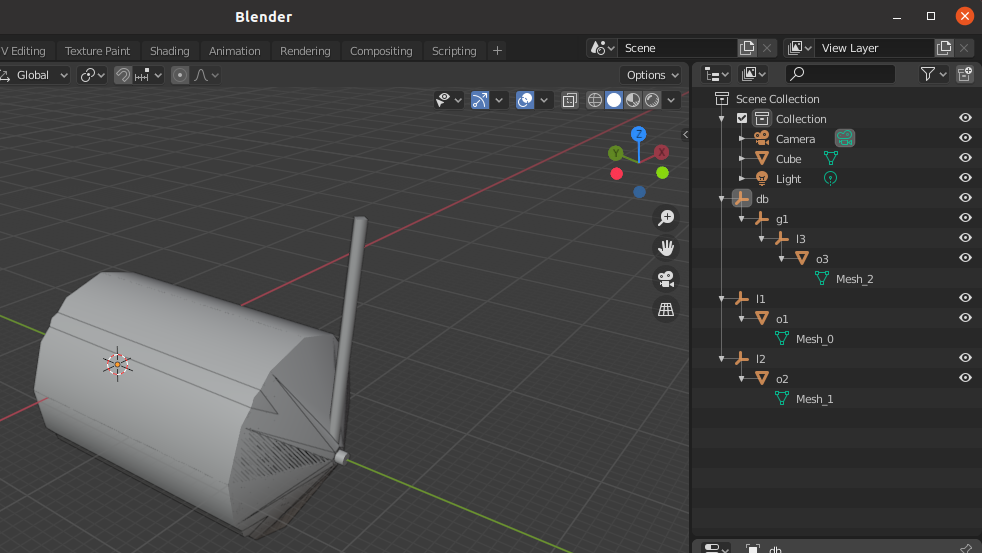
3 LOD nodes

Main group node

DB node



glTF import in Blender



* LOD nodes (Extension) are not supported in Blender – imported as just a node, with all 3 LODs visible at the same time (but could be separated as they are in different nodes)
* No node attribution

9 Sept 2020:

* Converter tool updated to support textures
* Support for CDB extension has started

