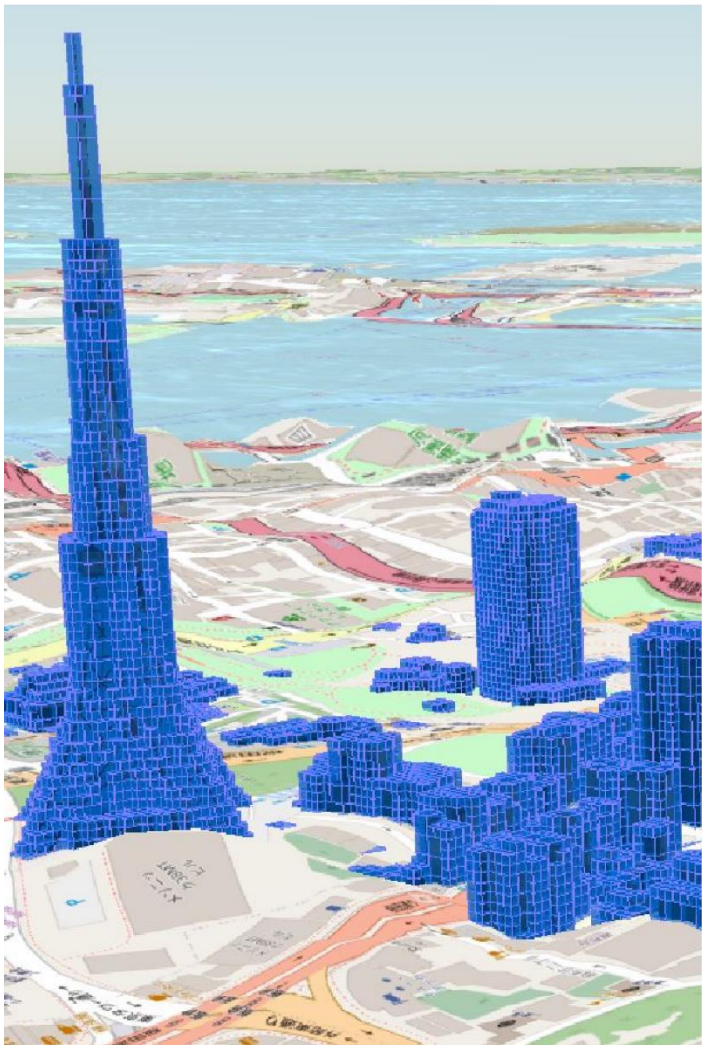
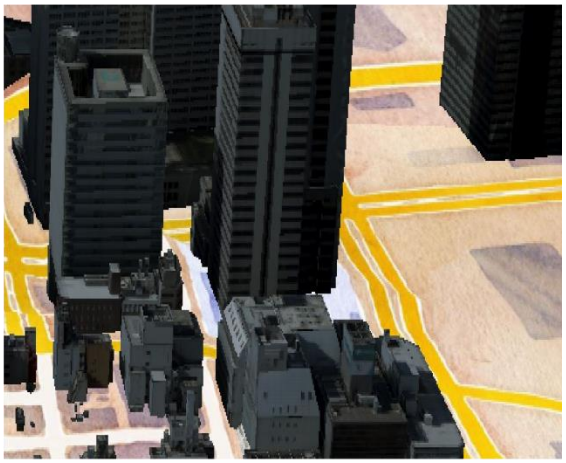
PLATEAU のための空間 ID 生成ツール 技術検証レポート

Technical Report for Spatial ID Generator on PLATEAU





はじめに

- デジタル庁は、2022 年度から国土交通省や経済産業省をはじめとする関係省庁と連携して、3D 都市モデルを含めた 3 次元空間情報等を統合的に流通させる基盤となる「3 次元空間情報基盤」の検討・構築を行っている。この「3 次元空間情報基盤」は、人・機械が一意に空間を特定するための 3 次元グリッド（ボクセル）識別子である「空間 ID」を検索キー（インデックス）として導入し、さまざまな空間情報を結合・検索することによって、自律移動ロボットをはじめとする様々なソリューションの基盤となるデジタルツインの社会実装を目指している。

- 国土交通省では、2020 年度から建築物や道路、設備など都市空間を構成する地物をデジタルツイン上で再現する 3D 都市モデルの整備・活用・オープンデータ化の取組みである Project PLATEAU（プラトー）に取り組んでおり、全国の地方公共団体においてデータ整備が進められている。3D 都市モデルは、「空間 ID」の普及・発展に欠かすことのできない空間情報の一つである。

- 2022 年度の本プロジェクト「デジタルツイン構築に向けた 3D 都市モデルの整備に関する調査研究」では、「3D 都市モデル標準製品仕様書 第 2.3 版」に準拠する 3D 都市モデルに対し「空間 ID」を自動付与するツールと、「空間 ID」のメタデータに 3D 都市モデルの持つ属性情報を自動付与するためのツールを開発するとともに、その標準化を図るため、標準仕様の拡張仕様（案）を作成した。

- 本ドキュメントは、上述のツールに関する技術仕様と、開発過程及び成果から得られた「空間 ID」活用に関する実証結果についてとりまとめたものである。また、本作成実証で使用した「空間 ID 付与のための 3D 都市モデル拡張製品仕様書（案）」の解説も示す。

- Project PLATEAU の 3D 都市モデルを「空間 ID」へ変換する際には、本ドキュメントで紹介するツールを活用し、「3 次元空間情報基盤」の構築に役立てていただきたい。

目次

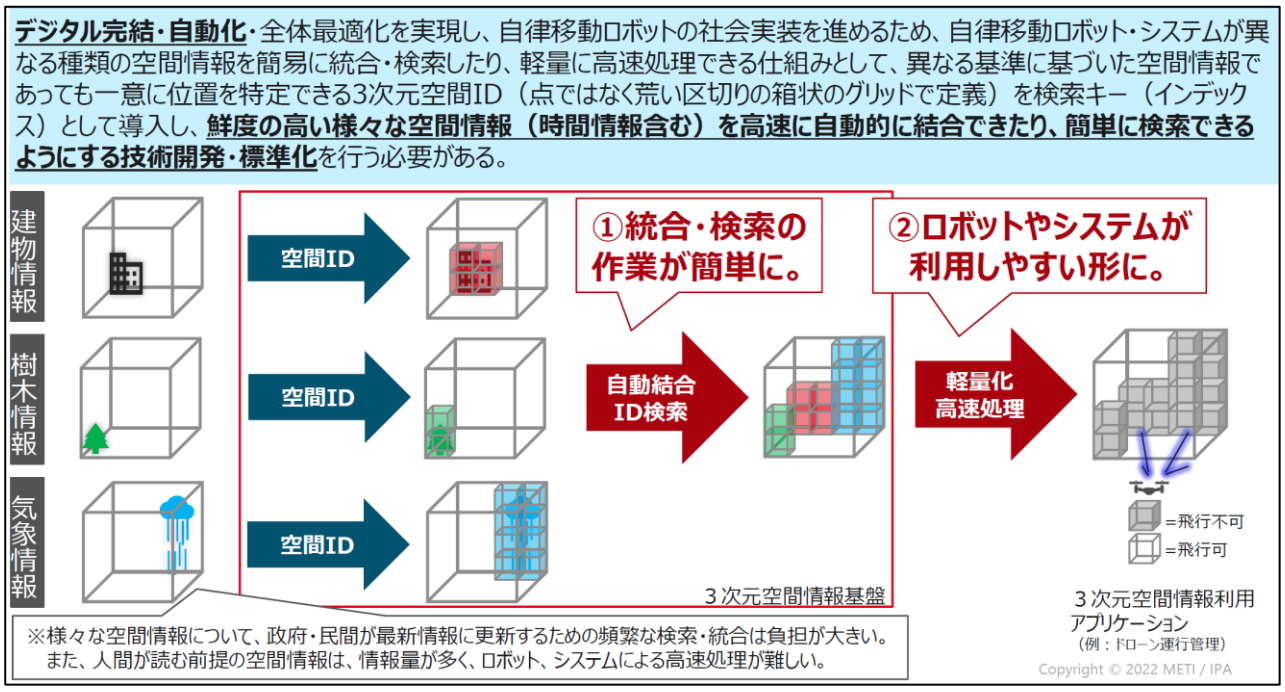
1 PLATEAU のための空間 ID 生成ツールの概要

(1) 開発の目的

デジタル庁は、国土交通省や経済産業省をはじめとする関係省庁と連携して、自動運転車、ドローン及び自動配送ロボット等による、運行環境をリアルタイムで把握し経路決定を行うなどの高度な運行を可能とするために、3D 都市モデル等のデジタルインフラ／デジタルツインの整備を進めている。

3D 都市モデルを含めた 3 次元空間情報等を統合的に流通させる基盤として「3 次元空間情報基盤」の検討・構築が進められており、その中で、実世界の位置情報を統一的な基準によって特定するための 3 次元グリッド（ボクセル）識別子である「空間 ID」が定義されている（図 1-1 参照）。

本自動生成ツールは、3 次元空間情報の重要な要素の一つであり、Project PLATEAU を通じて整備が進められている「3D 都市モデル標準製品仕様書 第 2.3 版」準拠の 3D 都市モデルに対し、「空間 ID」を自動付与する仕組みを用意することによって、「空間 ID」の実現・普及に貢献することを目的としている。



出典：3 次元空間情報基盤アーキテクチャ設計報告書

https://www.ipa.go.jp/dadc/architecture/pdf/pj_report_3dspatialinfo_doc_202208_1.pdf

図 1-1 「3 次元空間情報基盤」及び「空間 ID」の概要

本技術検証レポートは、「空間 ID を付与した 3D 都市モデル生成ツール」と「空間 ID のメタデータ生成ツール」に関する技術仕様、利用方法及び空間 ID の生成結果を記載する。また、作成実証で使用した「空間 ID 付与のための 3D 都市モデル拡張製品仕様書（案）」の解説も示す。

本技術検証レポートの作成は、次の業務で実施した。

- 業務名：デジタルツイン構築に向けた 3D 都市モデルの整備に関する調査研究
- 発注者：デジタル庁
- 受注者：株式会社パスコ・国際航業株式会社・中日本航空株式会社共同事業体
- 履行期間：令和 4 年 6 月 2 日 ～ 令和 5 年 3 月 24 日

(2) 開発したツールの全体像

この業務で開発したツールの全体像を図 1-2 に示す。

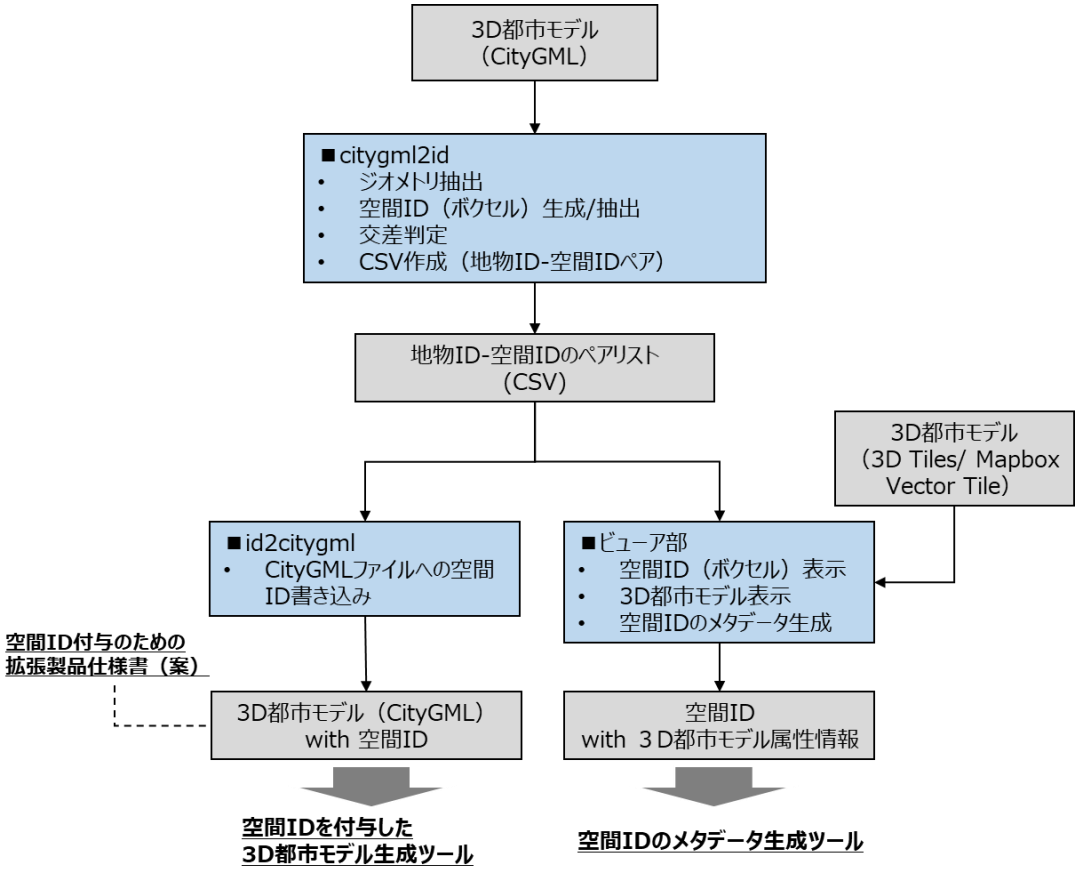


図 1-2 開発ツールの全体像

なお、開発したツールのソースコードは、Project PLATEAU の GitHub において、オープンソースとして公開されているので参照いただきたい。

- PLATEAU のための空間 ID 生成ツール
  ：https://github.com/Project-PLATEAU/PLATEAU-generator-for-spatialid

(3) 「空間 ID」の仕様

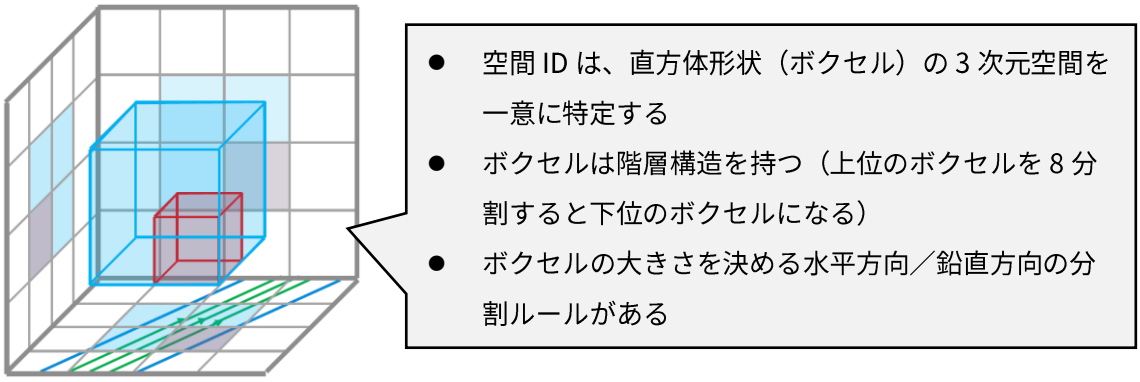
3D 都市モデルに付与する「空間 ID」は、独立行政法人情報処理推進機構に設置されたデジタルアーキテクチャ・デザインセンター（以下、「DADC」という）が設計する仕様に準拠する。DADC は、「自動移動ロボットプログラム※」を立ち上げ、その中に設置された「3 次元空間情報基盤アーキテクチャ検討会」にて、「空間 ID」に関する技術仕様を検討している。

※https://www.ipa.go.jp/dadc/architecture/wg_autonomousmobilerobot.html

「空間 ID」とは、実世界（3 次元空間）を、階層構造を持つ空間ボクセルで分割し、個々のボクセルに一意に与えられた識別子のことをいう。空間 ID のコンセプトとそのイメージを表 1-1 と図 1-3 に示す。

表 1-1　空間 ID のコンセプト

コンセプト		説明
1	ボクセルによる分割と ID の付与	3 次元空間を直方体形状のボクセルに分割する。
		各ボクセルは一意の ID（識別子）である「空間 ID」を持つ。
2	ボクセルは階層構造を持つ	階層構造（複数のズームレベル）を持ち、ズームレベルに応じてボクセルサイズが変化する。
		最上位の階層をズームレベル 0 とし、ズームレベルが 1 つ増えるごとにボクセルの 8 分割を繰り返す。
		階層（ズームレベル）間で親子関係を持つ。
		同一ズームレベルにおけるボクセル同士は重複しない。



出典：３次元空間情報基盤アーキテクチャ設計報告書（イメージ図を引用）

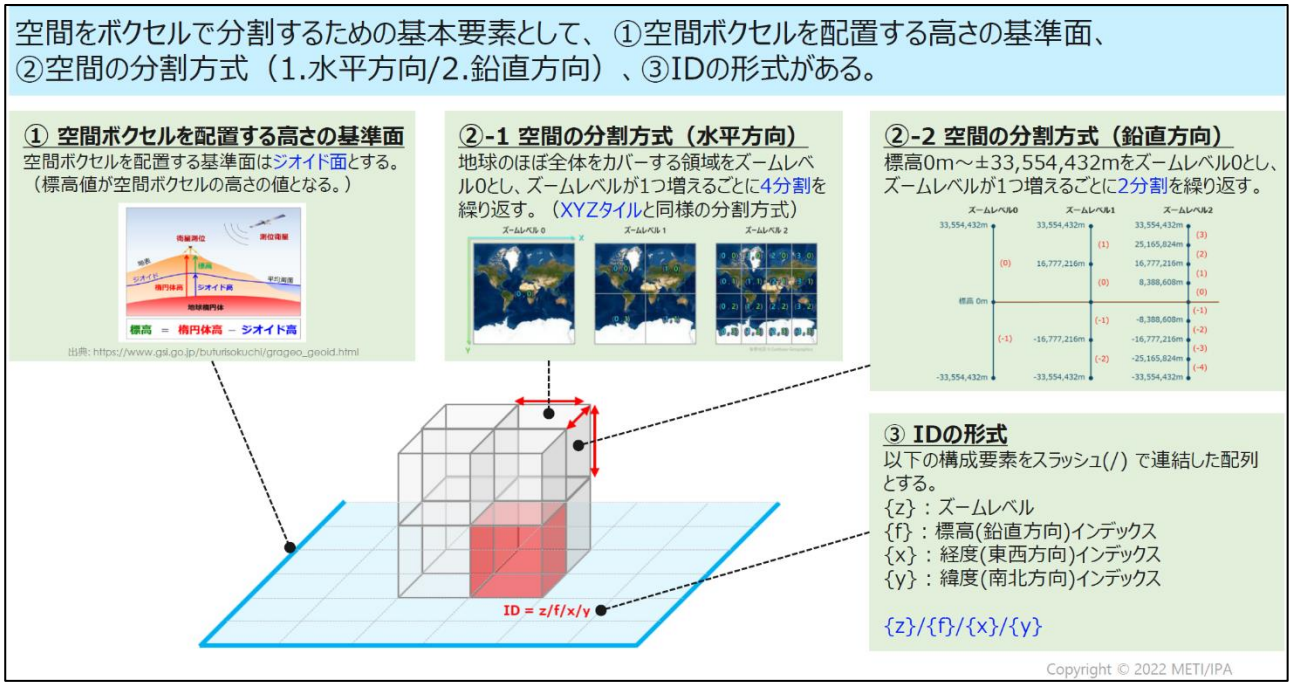
https://www.ipa.go.jp/dadc/architecture/pdf/pj_report_3dspatialinfo_doc_202208_1.pdf

図 1-3　空間 ID（空間ボクセル）のイメージ

「空間 ID」が定める空間（直方体形状のボクセル空間）の定義を表 1-2 に、構成要素の全体像を図 1-4 に示す。

表 1-2　空間 ID によって定まる空間ボクセルの定義

空間 ID によって定まる空間ボクセルの定義			
1	空間の分割方式	水平方向	地球のほぼ全体をカバーする領域をズームレベル 0 とし、ズームレベルが 1 つ増えるごとに 4 分割を繰り返す。
			地理院タイル（XYZ タイル）と同じ分割方式。https://maps.gsi.go.jp/development/siyou.html を参照。
		鉛直方向	標高 0m～±33,554,432m をズームレベル 0 とし、ズームレベルが 1 つ増えるごとに 2 分割を繰り返す。
			空間ボクセルを配置する高さの基準面はジオイド面。
2	空間 ID の形式		{z}/{f}/{x}{y}で記録される。{z}:ズームレベル、{f}:鉛直方向インデックス、{x}:経度(東西方向)インデックス、{y}:緯度(南北方向)インデックス



出典：第 5 回 3 次元空間情報基盤アーキテクチャ検討会資料

https://www.ipa.go.jp/dadc/architecture/pdf/pj_report_3dspatialinfo_doc-appendix_202212_1.pdf

図 1-4　空間 ID 構成要素の全体像

経度、緯度、標高及びズームレベルから空間 ID を構成する各インデックス（{f}、{x}、{y}）を算出するための計算式を次に示す。

lng：経度[10 進度]
lat：緯度[10 進度]、lat_rad：緯度[ラジアン]
h：標高[m]
z：ズームレベル

n = 2 ^ z
Z = 25（ボクセルの高さが 1m となるズームレベル）
H = 2 ^ Z[m]

f = floor(n * h / H)
x = floor(n * ((lng + 180) / 360))
y = floor(n * (1 - log(tan(lat_rad) + (1 / cos(lat_rad))) / PI) / 2)

出典：https://github.com/unvt/zfxy-spec

緯度 0 度の場合の各ズームレベルのボクセルサイズを、表 1-3 に示す。

表 1-3　ボクセルサイズの例（緯度 0 度の場合のサイズ）

ズームレベル	水平方向		鉛直方向	ズームレベル	水平方向		鉛直方向
	東西方向(m)	南北方向(m)	(m)		東西方向(m)	南北方向(m)	(m)
0	40,075,016.68	40,075,016.68	33,554,432	14	2,445.98	2,445.98	2,048
1	20,037,508.34	20,037,508.34	16,777,216	15	1,222.99	1,222.99	1,024
2	10,018,754.17	10,018,754.17	8,388,608	16	611.50	611.50	512
3	5,009,377.09	5,009,377.09	4,194,304	17	305.75	305.75	256
4	2,504,688.54	2,504,688.54	2,097,152	18	152.87	152.87	128
5	1,252,344.27	1,252,344.27	1,048,576	19	76.44	76.44	64
6	626,172.14	626,172.14	524,288	20	38.22	38.22	32
7	313,086.07	313,086.07	262,144	21	19.11	19.11	16
8	156,543.03	156,543.03	131,072	22	9.55	9.55	8
9	78,271.52	78,271.52	65,536	23	4.78	4.78	4
10	39,135.76	39,135.76	32,768	24	2.39	2.39	2
11	19,567.88	19,567.88	16,384	25	1.19	1.19	1
12	9,783.94	9,783.94	8,192	26	0.60	0.60	0.5
13	4,891.97	4,891.97	4,096				

出典：第 5 回 3 次元空間情報基盤アーキテクチャ検討会資料（表を引用）

https://www.ipa.go.jp/dadc/architecture/pdf/pj_report_3dspatialinfo_doc-appendix_202212_1.pdf

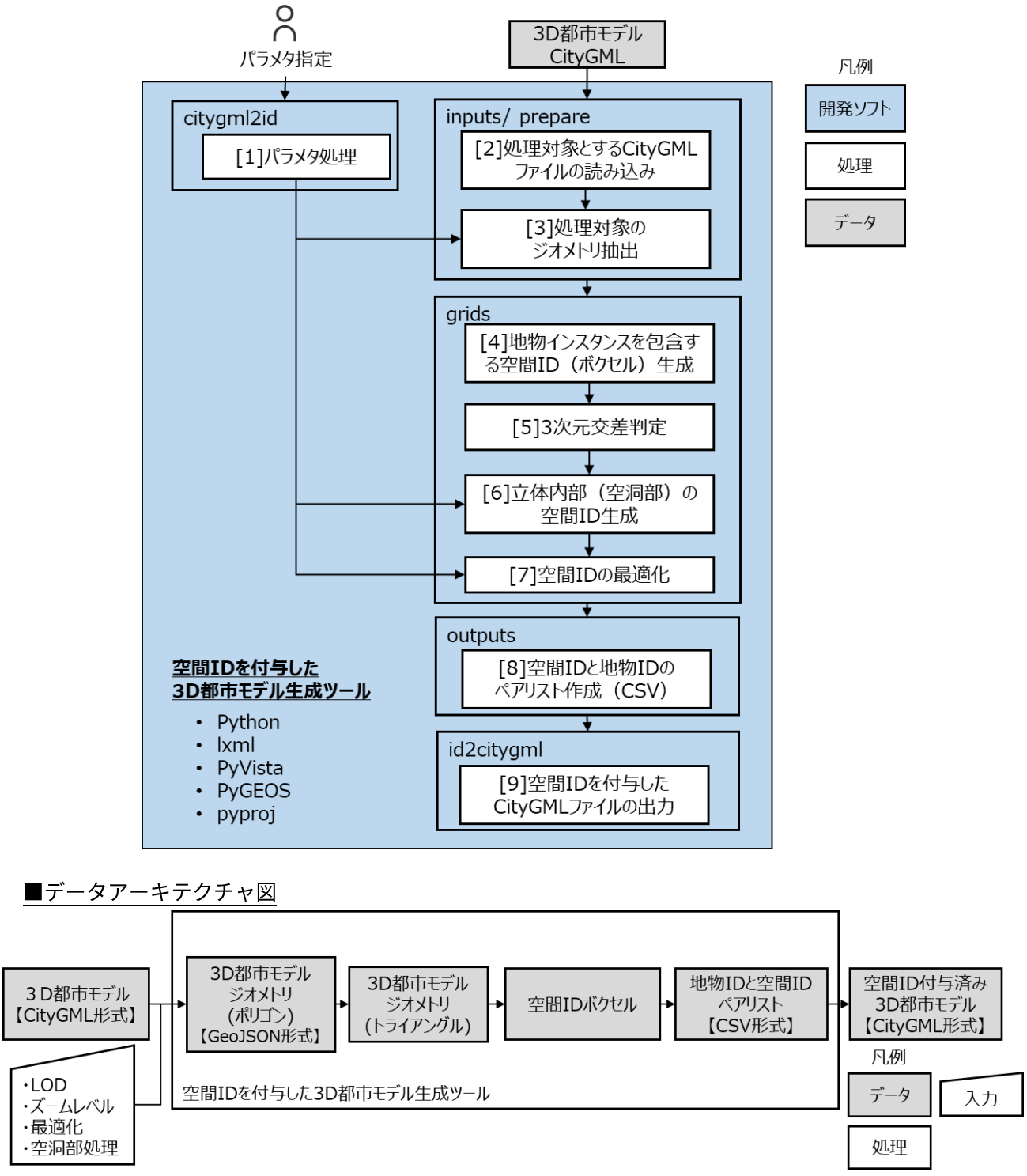
2 開発ツールの技術仕様

この業務で開発した「空間 ID を付与した 3D 都市モデル生成ツール」と「空間 ID のメタデータ生成ツール」の技術仕様を解説する。なお、両者のツールで使用したオープンソースソフトウェア（OSS）ライブラリは、商用利用される可能性も想定し、非コピーレフトのライセンスとした。

2.1 アーキテクチャ図

(1) 空間 ID を付与した 3D 都市モデル生成ツール

■システムアーキテクチャ図

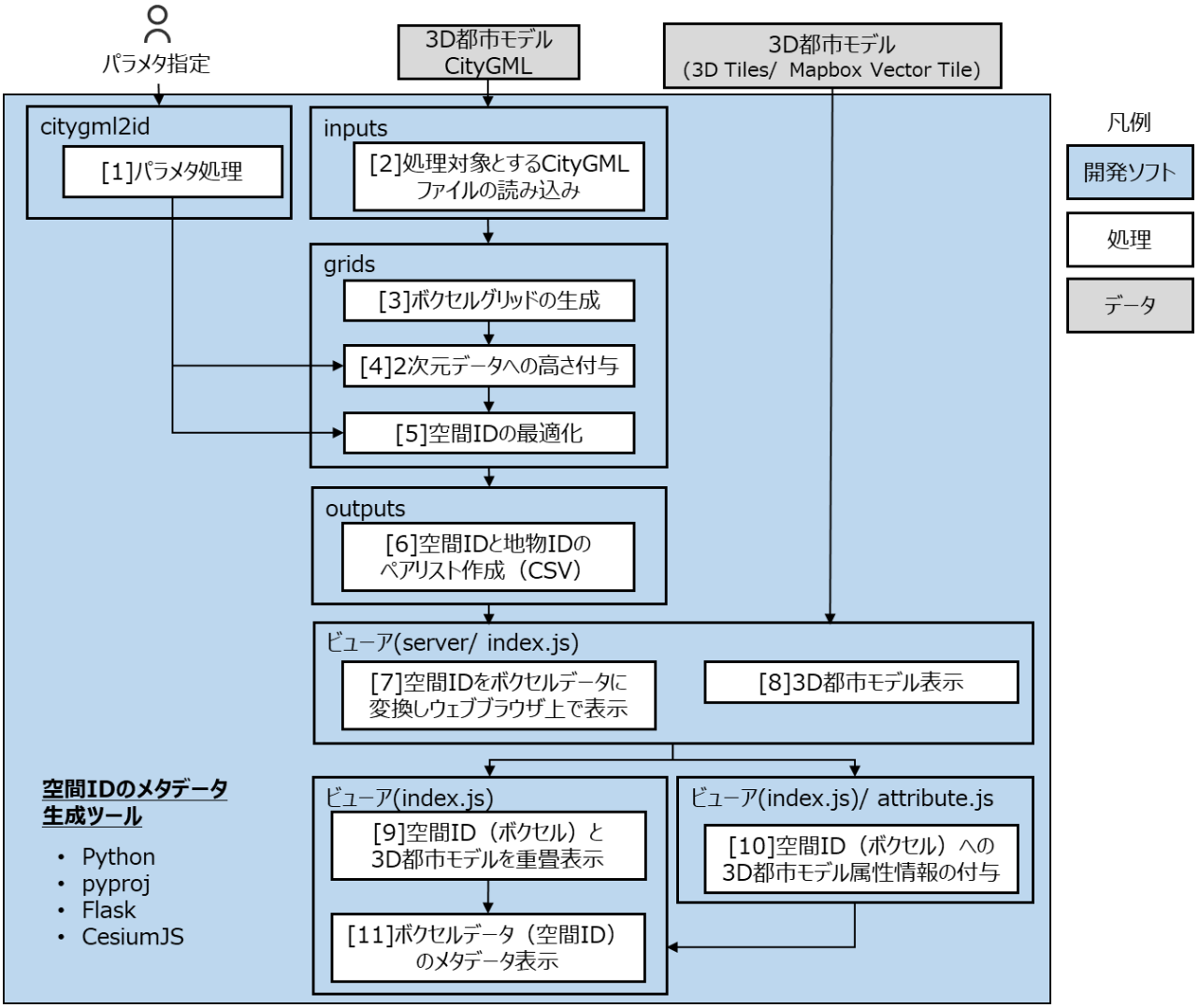


■データアーキテクチャ図

図 2-1 空間 ID を付与した 3D 都市モデル生成ツール：アーキテクチャ図

(2) 空間 ID のメタデータ生成ツール

■システムアーキテクチャ図



■データアーキテクチャ図

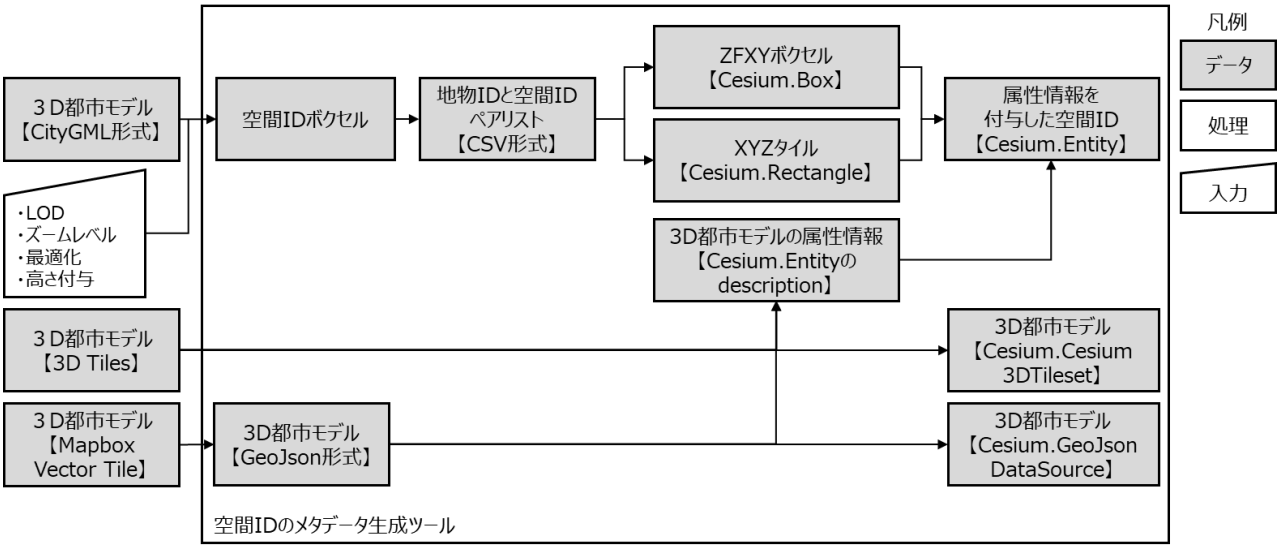


図 2-2　空間 ID のメタデータ生成ツール：アーキテクチャ図

2.2　システム機能設計

開発ツールのシステム機能を以降に示す。

(1) 空間 ID を付与した 3D 都市モデル生成ツール

1)　機能概要

本ツールは、コマンド処理によって動作するプログラムとし、バッチファイルやシェルスクリプトによって一括処理できるものとした。本ツールの機能概要を表 2-1 に示す。

表 2-1　空間 ID を付与した 3D 都市モデル生成ツールの機能概要

機能		説明	Python モジュール
1	パラメータ処理	●処理の基準とする空間ボクセルのズームレベル（最大ズームレベル）をパラメータで指定する。 ●交差判定の対象とするジオメトリの LOD をパラメータで指定する。パラメータ未指定の場合は LOD3 とする。 ●空洞部の空間 ID を生成するかどうかをパラメータで指定する。 ●空間 ID を最適化するかどうかをパラメータで指定する。	citygml2id
2	処理対象とする CityGML ファイルの読み込み	●CityGML ファイルを lxml を使ってロードする。 ●ジオメトリが XLink で別の要素を参照している場合は参照先のジオメトリを展開し、XLink を含まない一時的な XML ファイルに出力する。	inputs prepare
3	処理対象ジオメトリの抽出	●機能 2 で出力した XML ファイルを lxml を使ってロードする。 ●各地物に対して機能 1 で指定された LOD 以下の最も詳細度の高い LOD のジオメトリのみを処理対象とする。例えば、建築物のインスタンスが LOD2 と LOD1 のジオメトリをもつ場合、パラメタで LOD2 指定したときは、LOD2 のジオメトリのみを対象とする。 ●表 2-2 に示すように LOD に対応するジオメトリの要素名は地物クラスによって異なるため外部ファイル（resources/geometry.csv）にて定義する。このファイルではジオメトリの要素名に加え、ジオメトリが２次元であるか３次元であるかも定義する。抽出したジオメトリにもこの次元情報を保持しておき、機能 5 で参照する。 ●各地物の処理対象 LOD のジオメトリをポリゴンに分解する。Solid の場合は外側のサーフェスのみを対象とする。 ●pyproj を使って分解したポリゴンを擬似メルカトル座標（EPSG:3857）に変換する。	Inputs prepare

機能		説明	Python モジュール
		● １ポリゴン１地物となるように一時的な GeoJSON ファイルを作成する。その際、CityGML の gml:id 属性を GeoJSON の gml_id プロパティに設定する。よって、同じ gml_id プロパティ値を持つ地物が複数出力されることになる。 ● 一時的な GeoJSON を Python の標準ライブラリ json を使ってロードする。	
4	地物インスタンスを包含する空間 ID（ボクセル）の生成	PyVista を使ってポリゴンをトライアングルに分割し、その BoundingBox を算出する。 ● ZFXY タイルの水平方向については、機能 1 で指定されたズームレベルの XYZ タイルのうち、この BoundingBox を包含する最小の座標範囲を求め、対応するタイルインデックス範囲を計算する。 ● ZFXY タイルの鉛直方向については、BoundingBox を包含する最小座標範囲を求め、表 1-2 に従い対応するインデックス範囲を計算する。 ● 求めた ZFXY タイルの各要素から空間 ID の文字列を生成する。 ● 生成した空間 ID は地物 ID（gml:id）ごとに管理する。 ● 空間 ID（ボクセル）生成の際の座標計算については表 2-6 に詳述する。	grids
5	3 次元交差判定	● BoundingBox を包含する空間 ID 群から実際にトライアングル要素と交差する空間 ID を抽出する。その際、機能 3 で設定したジオメトリの次元情報を参照し、３次元あるいは２次元の処理を適用する。３次元の交差判定には PyVista ライブラリを、２次元の交差判定には PyGEOS ライブラリを使用する。 ● 具体的なアルゴリズムは表 2-6 に詳述する。	grids
6	立体内側（空洞部）の空間 ID の生成	● 鉛直方向の最小値と最大値の間を埋めることで Solid で表現される占有領域部分（Solid 内の空洞部）の空間 ID を生成する。 ● 空間 ID の文字列のフォーマットは表 1-2 に従う。ただし、２次元のデータは {z}/{x}/{y} に縮退したフォーマットとする。 ● この時点のズームレベルは機能1で指定したズームレベルである。 ● 機能 1 で空洞部の空間 ID を生成するように指定された場合のみ実行するオプション機能である。 ● 具体的なアルゴリズムは表 2-7 に詳述する。	grids
7	空間 ID の最適化	● 図 4-1 の考え方に基づき、空間 ID を上位のズームレベルに統合/最適化する。	grids

機能		説明	Python モジュール
		●機能 1 で空間 ID を最適化するように指定された場合のみ実行するオプション機能である。	
8	空間 ID と地物 ID のペアリスト生成（CSV 形式）	●機能 5、6 及び 7 の処理結果を CSV ファイルに出力する。 ●この CSV ファイルには表 4-5 で示すヘッダ行を出力する。	outputs
9	空間 ID を付与した CityGML ファイルの出力	●CityGML ファイルを、lxml ライブラリを使って読み込む。 ●機能 8 の出力である CSV ファイルから空間 ID と地物 ID のペアリストを読み込み、地物 ID をキーとし、空間 ID のリストを値とするディクショナリを作成する。 ●各地物インスタンスに対し、gml:id 属性の値でディクショナリを参照し、地物インスタンスに対応する空間 ID（複数）を取得する。 ●拡張製品仕様書（案）の符号化仕様に基づき、空間 ID 付与のための拡張属性要素を生成し、地物インスタンス要素内に挿入する。 ●拡張属性要素が追加された XML を CityGML ファイルに出力する。	Id2citygml

2)　空間 ID 付与対象の 3D 都市モデル

空間 ID 付与対象の地物と、3 次元交差判定に使用する空間属性（ジオメトリ）を表 2-2 に示す。

表 2-2　空間 ID 付与対象の地物と交差判定に使用する空間属性

地物			ジオメトリ（空間属性）
1	建築物	Building	lod3Solid
2			lod2Solid
3			lod1Solid
4			consistsOfBuildingPart (lod3Solid)
5			consistsOfBuildingPart (lod2Solid)
6			outerBuildingInstallation (lod3Geometry)
7			outerBuildingInstallation (lod2Geometry)
8	都市設備	CityFurniture	lod3Geometry
9			lod2Geometry
10			lod1Geometry
11	土地利用	LandUse	lod1MultiSurface
12	地形	ReliefFeature	tin
13	道路	Road	lod3MultiSurface

地物			ジオメトリ（空間属性）
14			lod2MultiSurface
15			lod1MultiSurface
16			lod0Network
17	植生	SolitaryVegetationObject	lod3Geometry
18			lod2Geometry
19			lod1Geometry
20		PlantCover	lod3MultiSolid
21			lod2MultiSolid
22			lod1MultiSolid
23			lod3MultiSurface
24			lod2MultiSurface
25			lod1MultiSurface
26	水部	WaterBody	lod1MultiSurface
27	都市計画決定情報	UrbanPlanningArea	lod1MultiSurface
28		QuasiUrbanPlanningArea	lod1MultiSurface
29		AreaClassification	lod1MultiSurface
30		DistrictsAndZones	lod1MultiSurface
31		UseDistrict	lod1MultiSurface
32		SpecialUseDistrict	lod1MultiSurface
33		SpecialUseRestrictionDistrict	lod1MultiSurface
34		ExceptionalFloorAreaRateDistrict	lod1MultiSurface
35		HighRiseResidentialAttractionDistrict	lod1MultiSurface
36		HeightControlDistrict	lod1MultiSurface
37		HighLevelUseDistrict	lod1MultiSurface
38		SpecifiedBlock	lod1MultiSurface
39		SpecialUrbanRenaissanceDistrict	lod1MultiSurface
40		HousingControlArea	lod1MultiSurface
41		ResidentialEnvironmentImprovementDistrict	lod1MultiSurface
42		SpetialUseAttractionDistrict	lod1MultiSurface
43		FirePreventionDistrict	lod1MultiSurface
44		SpecifiedDisasterPreventionBlockImprovementZone	lod1MultiSurface
45		LandscapeZone	lod1MultiSurface
46		ScenicDistrict	lod1MultiSurface
47		ParkingPlaceDevelopmentZone	lod1MultiSurface
48		PortZone	lod1MultiSurface
49		SpecialZoneForPreservationOfHistoricalLandscape	lod1MultiSurface
50		ZoneForPreservationOfHistoricalLandscape	lod1MultiSurface
51		GreenSpaceConservationDistrict	lod1MultiSurface
52		SpecialGreenSpaceConservationDistrict	lod1MultiSurface
53		TreePlantingDistrict	lod1MultiSurface
54		DistributionBusinessZone	lod1MultiSurface
55		ProductiveGreenZone	lod1MultiSurface

地物			ジオメトリ（空間属性）
56		ConservationZoneForClustersOfTraditionalStructures	lod1MultiSurface
57		AircraftNoiseControlZone	lod1MultiSurface
58		ProjectPromotionArea	lod1MultiSurface
59		UrbanRedevelopmentPromotionArea	lod1MultiSurface
60		LandReadjustmentPromotionArea	lod1MultiSurface
61		ResidentialBlockConstructionPromotionAreas	lod1MultiSurface
62		LandReadjustmentPromotionAreasForCoreBusinessUrbanDevelopment	lod1MultiSurface
63		UnusedLandUsePromotionArea	lod1MultiSurface
64		UrbanDisasterRecoveryPromotionArea	lod1MultiSurface
65		UrbanFacility	lod1MultiSurface
66		TrafficFacility	lod1MultiSurface
67		OpenSpaceForPublicUse	lod1MultiSurface
68		SupplyFacility	lod1MultiSurface
69		TreatmentFacility	lod1MultiSurface
70		Waterway	lod1MultiSurface
71		EducationalAndCulturalFacility	lod1MultiSurface
72		MedicalFacility	lod1MultiSurface
73		SocialWelfareFacility	lod1MultiSurface
74		MarketsSlaughterhousesCrematoria	lod1MultiSurface
75		CollectiveHousingFacilities	lod1MultiSurface
76		CollectiveGovernmentAndPublicOfficeFacilities	lod1MultiSurface
77		DistributionBusinessPark	lod1MultiSurface
78		CollectiveFacilitiesForTsunamiDisasterPrevention	lod1MultiSurface
79		CollectiveFacilitiesForReconstructionAndRevitalization	lod1MultiSurface
80		CollectiveFacilitiesForReconstruction	lod1MultiSurface
81		CollectiveUrbanDisasterPreventionFacilities	lod1MultiSurface
82		UrbanFacilityStipulatedByCabinetOrder	lod1MultiSurface
83		TelecommunicationFacility	lod1MultiSurface
84		WindProtectionFacility	lod1MultiSurface
85		FireProtectionFacility	lod1MultiSurface
86		TideFacility	lod1MultiSurface
87		FloodPreventionFacility	lod1MultiSurface
88		SnowProtectionFacility	lod1MultiSurface
89		SandProtectionFacility	lod1MultiSurface
90		UrbanDevelopmentProject	lod1MultiSurface
91		LandReadjustmentProject	lod1MultiSurface
92		NewHousingAndUrbanDevelopmentProject	lod1MultiSurface
93		IndustrialParkDevelopmentProject	lod1MultiSurface
94		UrbanRedevelopmentProject	lod1MultiSurface
95		NewUrbanInfrastructureProject	lod1MultiSurface

地物			ジオメトリ（空間属性）
96		ResidentialBlockConstructionProject	lod1MultiSurface
97		DisasterPreventionBlockImprovementProject	lod1MultiSurface
98		UrbanRenewalProject	lod1MultiSurface
99		ScheduledAreaForUrbanDevelopmentProject	lod1MultiSurface
100		ScheduledAreaForNewHousingAndUrbanDevelopmentProjects	lod1MultiSurface
101		ScheduledAreaForIndustrialParkDevelopmentProjects	lod1MultiSurface
102		ScheduledAreaForNewUrbanInfrastructureProjects	lod1MultiSurface
103		ScheduledAreaForCollectiveHousingFacilities	lod1MultiSurface
104		ScheduledAreaForCollectiveGovernmentAndPublicOfficeFacilities	lod1MultiSurface
105		ScheduledAreaForDistributionBusinessPark	lod1MultiSurface
106		DistrictPlan	lod1MultiSurface
107		RoadsideDistrictPlan	lod1MultiSurface
108		RuralDistrictPlan	lod1MultiSurface
109		HistoricSceneryMaintenanceAndImprovementDistrictPlan	lod1MultiSurface
110		DisasterPreventionBlockImprovementZonePlan	lod1MultiSurface
111		DistrictDevelopmentPlan	lod1MultiSurface
112		SpecifiedBuildingZoneImprovementPlan	lod1MultiSurface
113		DistrictImprovementPlanForDisasterPreventionBlockImprovementZonePlan	lod1MultiSurface
114		RoadsideDistrictImprovementPlan	lod1MultiSurface
115		RuralDistrictImprovementPlan	lod1MultiSurface
116		DistrictImprovementPlanForHistoricSceneryMaintenanceAndImprovementDistrict	lod1MultiSurface
117		PromotionDistrict	lod1MultiSurface
118		District	lod1MultiSurface
119		DistrictFacility	lod1MultiSurface
120		RuralDistrictFacility	lod1MultiSurface
121		RoadsideDistrictFacility	lod1MultiSurface
122		ZonalDisasterPreventionFacility	lod1MultiSurface
123	土砂災害警戒区域	SedimentDisasterProneArea	lod1MultiSurface

(2) 空間 ID のメタデータ生成ツール

1) 機能概要

本ツールは、3D 都市モデルから空間 ID を抽出し、抽出した空間 ID に 3D 都市モデルの属性を付与する。また、Web ブラウザ上に空間 ID（ボクセル）と 3D 都市モデルを表示し、3D 都市モデルの属性を、空間 ID のメタデータとして表示する。　本ツールの機能概要を、表 2-3 に示す。

表 2-3　空間 ID のメタデータ生成ツールの機能

機能		説明	Python モジュール / JavaScript ファイル
1	パラメータ処理	●空間 ID を最適化するかどうかをパラメータで指定する。 ●２次元データに高さを付与する場合、高さの範囲をパラメータで指定する。	citygml2id
2	処理対象とする CityGML ファイルの読み込み	●CityGML ファイルを lxml を使ってロードする。	inputs
3	ボクセルグリッドの生成	●CityGML に付与された空間 ID を抽出しボクセルグリッドを構築する。 ●空間 ID が {z}/{x}/{y} で表される２次元の場合 {f} に相当する要素には None を設定し、３次元のデータと区別できるようにしておく。	grids
4	2 次元データへの高さ付与	●機能１で指定された最小高さから最大高さまで３次元空間 ID を生成する。 ●機能１で高さを付与するように指定された場合のみ実行するオプション機能である。 ●2 次元地物への空間 ID 付与の考え方の補足説明を 2) に示す。	grids
5	空間 ID の最適化	●図 4-1 の考え方に基づき、空間 ID を上位のズームレベルに統合/最適化する。 ●機能 1 で空間 ID を最適化するように指定された場合のみ実行するオプション機能である。	grids
6	空間 ID と地物 ID のペアリスト生成（CSV 形式）	●機能 3、4 及び 5 の処理結果を CSV ファイルに出力する。 ●この CSV ファイルには表 4-5 で示すヘッダ行を出力する。	outputs
7	空間 ID をボクセルデータに変換し Web ブラウザ上で表示	●地物 ID と空間 ID のペアリストを記録した外部ファイル（CSV 形式：図 4-4 参照）をユーザがアップロードする。	server (ビューア) index.js（ビューア)

機能		説明	Python モジュール / JavaScript ファイル
		●アップロードされたファイルを受け取ったサーバは空間 ID からボクセル座標を計算する。 ●Web ブラウザでのボクセル表示に Cesium.Box クラスを使用するため、空間 ID からボクセル中心の座標と３辺のサイズを EPSG:4978 で求める。 ●ボクセルの高さを求める場合、標高に EGM96 のジオイド高を付加する。座標計算のアルゴリズムは 2.3(2)の 5)空間 ID（ボクセル）の座標計算アルゴリズム に詳述する。 ●各ボクセルに対して中心座標、サイズ、空間 ID、地物 ID（複数）を関連づけて JSON 形式で Web ブラウザに返却する。 ●JSON を受け取ったクライアントは Cesium.Box でボクセルジオメトリを構築し Cesium.Entity として表示する。 ●２次元データには Cesium.Rectangle を使い、Cesium.Entity として表示する。 ●Cesium.Entity には gml_ids という名前で当該ボクセルに含まれる地物 ID の配列を格納しておく。	
8	3D 都市モデルの表示	● CityGML から変換された PLATEAU View 用の 3D Tiles/MVT（Mapbox Vector Tile）をアップロードする。アップロードするファイルは事前に ZIP アーカイブしておく。 ●3D Tiles を Web 配信できるようにサーバの所定のディレクトリに配置する。 ●MVT を GeoJSON に変換する。MVT に複数ズームレベルのデータが含まれている場合は最大ズームレベルのデータのみを変換する。変換した GeoJSON は Web 配信できるようにサーバの所定のディレクトリに配置する。 ●3D Tiles/MVT(GeoJSON)を Web ブラウザ上に表示する。3D Tiles は Cesium.Cesium3DTileset を使って表示する。MVT（GeoJSON）は Cesium.GeoJsonDataSource を使って表示する。 ●この機能は 3D 都市モデルとの重畳表示により空間 ID	server (ビューア) index.js（ビューア)

機能		説明	Python モジュール / JavaScript ファイル
		（ボクセル）の妥当性を検証するための副次的な機能である。	
9	空間 ID（ボクセル）と 3D 都市モデルを重畳表示	●Web ブラウザ上で同時に表示する。 ●チェックボックスによりデータの表示を切り替えられるようにする。	index.js（ビューア)
10	空間 ID（ボクセル）への 3D 都市モデル属性情報の付与	●Cesium.Entity 形式の空間 ID（ボクセル）に格納された gml_ids の各値に対応する属性情報を 3D Tiles/MVT(GeoJSON) から取得し Cesium.Entity の description プロパティに HTML 形式で設定する。 ●3D Tiles / MVT(GeoJSON) が保持する属性情報のうち空間 ID（ボクセル）に付与する属性項目は attribute.js の attribute_def_3d および attribute_def_2d に定義する。設定方法については 2.3(2)の 6)属性付与の設定 に詳述する。 ●この機能は 3D 都市モデル、空間 ID（ボクセル）の順にデータをアップロードした際に実行され、逆の順序でアップロードした場合は実行されない。 ●この機能は 3D 都市モデルとの重畳表示により空間 ID（ボクセル）の妥当性を検証するための副次的な機能である。	index.js（ビューア) attribute.js（ビューア)
11	ボクセルデータ（空間 ID）のメタデータ表示	● Cesium.Entity の description に設定した属性情報の表示イメージを 図 2-3 に示す。	index.js（ビューア)

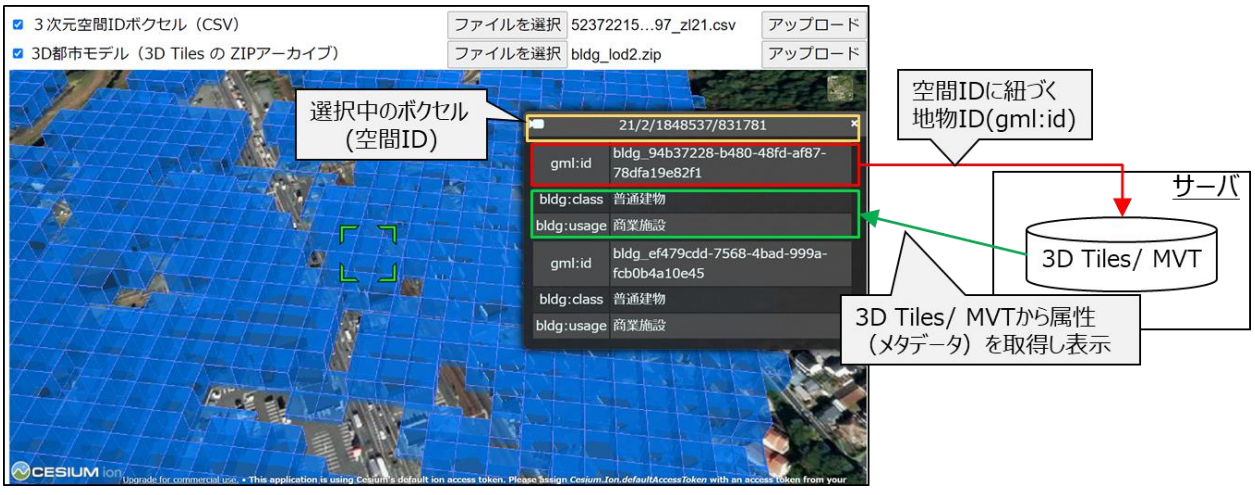
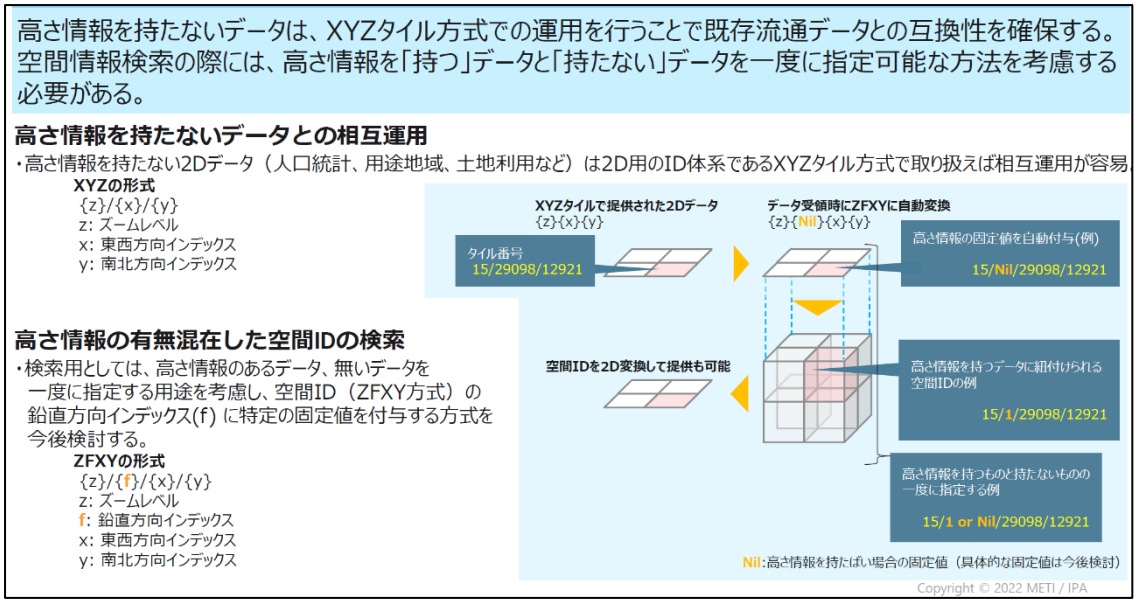


図 2-3　空間 ID のメタデータ表示の仕組み

2)　2 次元地物への空間 ID 付与の考え方

高さ情報を持たない 2 次元の地物に対応した空間 ID の形式については、図 2-4 に示すとおり、地理院タイル（XYZ タイル）へ変換することが検討されている。

一方で、高さを持たない面として整備される 3D 都市モデルの土地利用や都市計画決定情報を、地上/地下の空間を指す高さを持つボクセルに反映し活用するケースも想定される。そこで、本ツールでは、2 次元の地物に付与された 2 次元の空間 ID（XYZ タイル）から、指定した標高値の範囲内の 3 次元の空間 ID（ZFXY タイル）を生成する機能を用意した。変換の流れを図 2-5 に示す。



出典：3 次元空間情報基盤アーキテクチャ設計報告書

https://www.ipa.go.jp/dadc/architecture/pdf/pj_report_3dspatialinfo_doc_202208_1.pdf

図 2-4　高さ情報を持たないデータに対応した空間 ID の形式

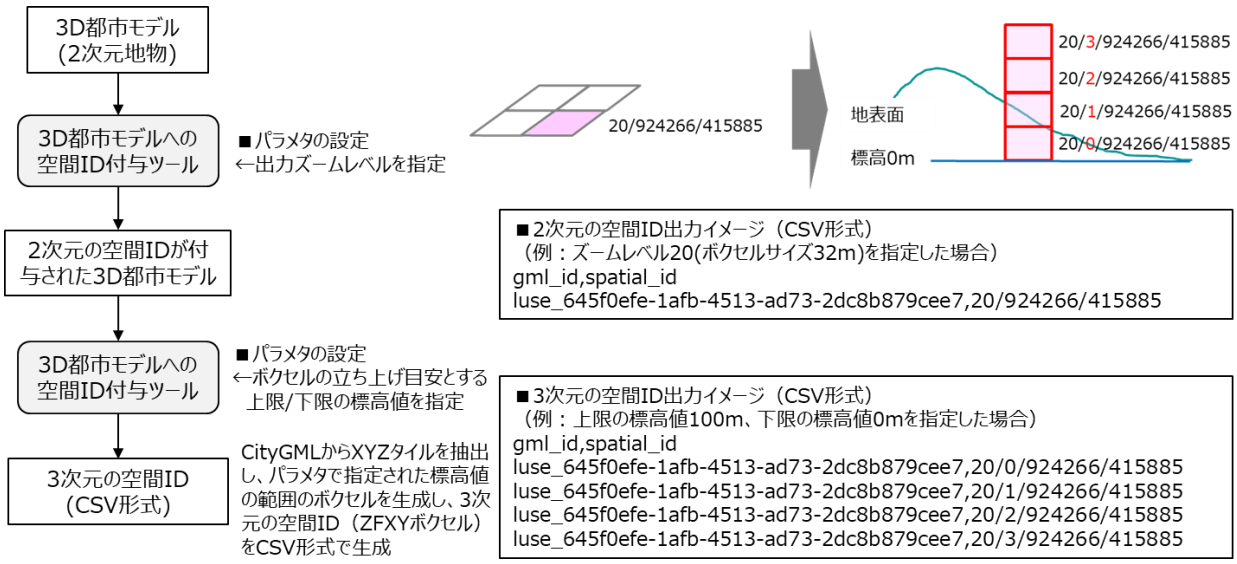


図 2-5　2 次元地物の空間 ID（XYZ タイル）を 3 次元の空間 ID(ZFXY タイル)へ変換する流れ

3)　ビューアで表示する 3D 都市モデル

ビューアで表示する 3D 都市モデルは、FME Hub で公開されている FME ワークスペース「PLATEAU2 可視化用データ変換（https://hub.safe.com/?page=1&page_size=10&order=relevance&query=plateau）」によって変換された Cesium 3D Tiles データセット (3D モデル) または Mapbox Vector Tile (MVT) データセット (2D ポリゴン)を使用する。メタデータの表示には gml:id を使用するため、JSON 属性に gml:id が記録されていない場合は、このワークスペースを編集し出力する。

2.3 プログラム設計

(1) 空間 ID を付与した 3D 都市モデル生成ツール

1) プログラム構成

「空間 ID を付与した 3D 都市モデル生成ツール」のプログラム構成を、表 2-4 に示す。

表 2-4 空間 ID を付与した 3D 都市モデル生成ツールのプログラム構成

プログラム構成			説明
1	メインプログラム	citygml2id.py	●CityGML から gml_id と空間 ID のペアを CSV 出力するツール ●表 2-1 に示す機能 1～8 に対応
2		id2citygml.py	●空間 ID の CSV ファイルから CityGML ファイルを更新（空間 ID の拡張属性作成）するツール ●表 2-1 の機能 9 に対応
3	サブプログラム	grids.py	●ボクセルグリッドを構築するためのプログラム
4		inputs.py	●ファイルを入力するためのプログラム
5		outputs.py	●ファイルを出力するためのプログラム
6		prepare.py	●CityGML ファイルを前処理（解析）するためのプログラム
7		constants.py	●定数定義

表 2-4 に示す各プログラムの関数構成を表 2-5 に示す。

表 2-5 プログラム（空間 ID を付与した 3D 都市モデル生成ツール）の関数構成

プログラム		関数名		説明
1	citygml2id.py	1-1	main	●[extract]のフラグに応じ、以下の関数を切り替えて処理 [1-2] xml2id [1-3] geom2id ●以下の関数を内部参照 [3-1]grids.get [1-2] xml2id [1-3] geom2id
		1-2	xml2id	●空間 ID 付与済の CityGML ファイルから、拡張属性に記録されている空間 ID を抽出 ●以下の関数を内部参照 [4-1] inputs.load_xml [3-2-15] grid.extract_ids [3-2-16] grids.extrude [5-1] outputs.export_csv

プログラム		関数名		説明
		1-3	geom2id	●ジオメトリから 3 次元交差判定を用いて、空間 ID を生成 ●以下の関数を内部参照 [4-2] inputs.load_features [3-2-4] grid.load_geom_data [5-1] outputs.export_csv
2	id2citygml.py	2-1	main	●以下の関数を内部参照 [4-4] inputs.get_target_gml_files [4-5] inputs.get_target_id_files [5-2] outputs.build_output_paths [2-2] find_id_file [2-8] get_output_element_info_file [4-1] inputs.load_xml [2-3] load_ids [2-4] update_xml [2-7] output_xml
		2-2	find_id_file	●.gml ファイルに対応する.csv ファイルの検索
		2-3	load_ids	●CSV ファイルから空間 ID を読み込み ●以下の関数を内部参照 [4-3] inputs.load_ids
		2-4	update_xml	●CityGML を更新 ●以下の関数を内部参照 [2-5] update_xml_embedding [2-6] update_xml_reference
		2-5	update_xml_embedding	●空間 ID を CityGML 拡張属性に埋め込み ●以下の関数を内部参照 [2-10] get_output_element_info
		2-6	update_xml_reference	●外部ファイル（CSV ファイル）参照用の相対パスを CityGML 拡張属性に埋め込み ●以下の関数を内部参照 [2-9] get_output_element_info_doc
		2-7	output_xml	●CityGML ファイルを出力

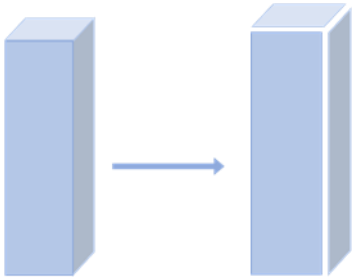
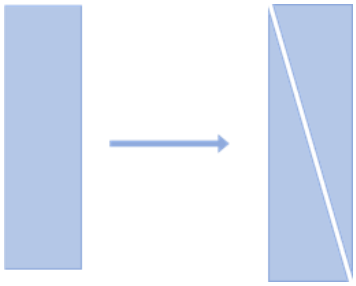
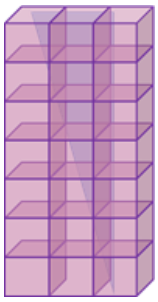
プログラム		関数名		説明
		2-8	get_output_element_info_file	●CityGML ファイル内の地物型から空間 ID の出力要素の情報を取得 ●以下の関数を内部参照 [4-1] inputs.load_xml [2-9] get_output_element_info_doc
		2-9	get_output_element_info_doc	●メモリ上の地物型の情報から空間 ID の出力要素の情報を取得 ●以下の関数を内部参照 [2-10] get_output_element_info
		2-10	get_output_element_info	●空間 ID の出力要素の情報をロードし取得
3	grids.py	3-1	get	●ボクセルグリッドを生成
		3-2	Grid	●ボクセルグリッドオブジェクトのクラス定義
		3-2-1	normalize_size	●ボクセルサイズの正規化
		3-2-2	encode_key	●座標インデックスを空間 ID へ変換
		3-2-3	decode_key	●空間 ID を座標値へ変換
		3-2-4	load_geom_data	●ジオメトリデータの読み込み(PyVista によるポリゴンデータ処理)
		3-2-5	_update_by_geom_3d (_update_by_geom_2d)	●ジオメトリデータを用いたアップデート(PyVista /PyGEOS を用いたポリゴンデータ処理)
		3-2-6	build_collision_filter	●交差判定(PyVista による衝突判定)の準備
		3-2-7	build_geom_3d	●対象オブジェクトを構成するポリゴンジオメトリの作成(PyVista によるポリゴンデータ処理)
		3-2-8	build_box_3d	●3 次元交差判定のためのボクセルジオメトリの作成(PyVista によるキューブデータ処理)
		3-2-9	build_geom_2d	●対象オブジェクトを構成するポリゴンジオメトリの作成(PyGEOS によるポリゴンデータ処理)
		3-2-10	build_box_2d	●2 次元交差判定のための矩形ジオメトリの作成(PyGEOS による矩形データ処理)
		3-2-11	interpolate	●空洞を埋めるための補間処理
		3-2-12	merge	●ボクセルを統合する最適化処理
		3-2-13	load_id_data	●空間 ID の読み込み
		3-2-14	_update_by_id	●空間 ID による更新

プログラム		関数名		説明
		3-2-15	extract_ids	●空間 ID の抽出
		3-2-16	extrude	●2 次元データへの高さ付与
		3-3	ZFXYGrid	●ZFXY グリッドのクラス定義
		3-3-1	encode_key	●座標インデックスを空間 ID へ変換
		3-3-2	decode_key	●空間 ID を座標値へ変換
		3-3-3	merge	●ボクセルを統合する最適化処理
		3-3-4	_merge_sub	●ボクセルを統合する最適化処理
		3-3-5	_update_by_id	●空間 ID による更新
		3-3-6	extract_ids	●空間 ID の抽出
		3-3-7	extrude	●2 次元データへの高さ付与
4	inputs.py	4-1	load_xml	●CityGML ファイルの読み込み
		4-2	load_features	●ジオメトリの読み込み
		4-3	load_ids	●空間 ID の読み込み
		4-4	get_target_gml_files	●ディレクトリ内の処理対象となる .gml ファイルの列挙
		4-5	get_target_id_files	●ディレクトリ内の処理対象となる .csv ファイルの列挙
5	outputs.py	5-1	export_csv	●地物 ID と空間 ID のペアリストを CSV 形式で出力
		5-2	build_output_paths	●ディレクトリ内の出力パスの一括生成
6	prepare.py	6-1	main	●入力となる CityGML ファイルの解析
		6-2	resolve_xlink	●XLink の解決
		6-3	extract_geometry	●ジオメトリの抽出
		6-4	get_geometry_paths	●ジオメトリ要素の XPath を取得
		6-5	arrange_path	●XML 要素の検索用に XPath を調整
		6-6	arrange_namespace	●XML 内の名前空間の宣言を調整
		6-7	arrange_namespace_file	●XML ファイル内の名前空間の宣言を調整

2)　3 次元の交差／占有領域判定のアルゴリズム

サブプログラム「grids.py」で行う 3D 都市モデルと空間 ID（空間ボクセル）の交差／占有領域の判定のアルゴリズムを表 2-6 と表 2-7 に示す。

表 2-6　3 次元の交差判定のアルゴリズム

	処理の流れ	イメージ
1	●Solid を Polygon に分解	
2	●Polygon を Triangle に分解 ●Triangle への分解には PyVista ライブラリの PolyData.triangulate() メソッドを利用する。	
3	●Triangle の Minimum Bounding Box と重なる ボクセルを抽出（ボクセルの頂点座標はズームレベルから決定する） ●ボクセルの頂点座標は擬似メルカトル座標系（EPSG:3857）で計算する。ズームレベル 0 のボクセルの水平方向の 1 辺のサイズは 2π*6378137 (m) である。ズームレベルが 1 増えるごとにサイズは 1/2 になるため、ズームレベル z のボクセルの水平方向の 1 辺のサイズ h(z) は以下のようになる。 h(z) = 2π*6378137 / 2 ^ z (m) 鉛直方向についてはズームレベル 0 のサイズが 2 ^ 25 (m) であるため、ズームレベル z のボクセルの鉛直方向の 1 辺のサイズ v(z) は以下のようになる。 v(z) = 2 ^ 25 / 2 ^ z ●Triangle の Minimum Bounding Box の座標範囲を min_x, max_x, min_y, max_y, min_z, max_z とすると、該当するボクセル（複数）の座標範囲は以下のようになる。 最小 X：floor(min_x / h(z)) * h(z) 最大 X：ceil(max_x / h(z)) * h(z) 最小 Y：floor(min_y / h(z)) * h(z) 最大 Y：ceil(max_y / h(z)) * h(z) 最小 Z：floor(min_z / v(z)) * v(z) 最大 Z：ceil(max_z / v(z)) * v(z)	

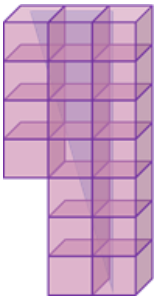
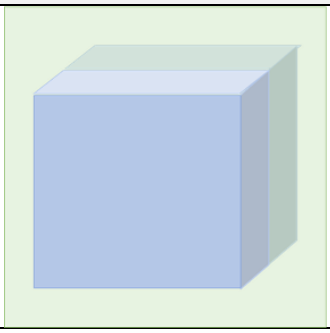
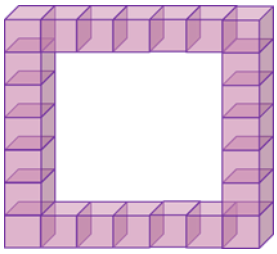
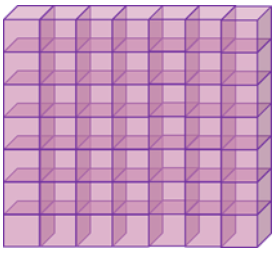
	処理の流れ	イメージ
4	●抽出したボクセル と Triangle が交差するかどうか判定 ●擬似メルカトル座標系の座標値は絶対値が大きいため、ボクセルサイズが小さいと浮動小数点の計算の過程で桁落ちが発生する可能性があるため、Triangle、ボクセルともに原点付近に平行移動し、絶対値を小さくしてから交差判定の計算を行う。 ●交差判定には PyVista ライブラリの vtkCollisionDetectionFilter クラスを利用する。サイズが同じで位置が異なる大量のボクセルが計算対象になることから、vtkCollisionDetectionFilter.SetMatrix() により作成済みボクセルを平行移動して再利用することで高速化を図る。	
5	●交差するすべてのボクセルをマージ（地物インスタンスごとに空間 ID を出力する必要があるため）	

表 2-7　3 次元の占有領域判定のアルゴリズム

	処理の流れ	イメージ
1	●表 2-6 の交差判定によりすべての Polygon に対して、ボクセルが作成されている状態からスタート ●説明のため Solid の断面に着目する	
2	●Polygon に対して交差判定したため、Solid の内部には ボクセルが作成されていない	
3	●鉛直方向の座標値が最小のボクセルと最大のボクセルの間にあるボクセルをすべて追加する	

(2) 空間 ID のメタデータ生成ツール

1) 全体構成

図 2-2 のアーキテクチャ図に示した通り、本ツールは空間 ID が付与された CityGML ファイルを入力とし、地物 ID と空間 ID のペアリストファイル（CSV 形式）を出力するコマンド部と、地物 ID と空間 ID のペアリストファイルを可視化するためのビューア部に分かれている。ビューア部はサーバ機能とクライアント機能に分かれているため、2)及び 3)に画面イメージとソフトウェア構成を示す。

2) ビューア部の画面イメージ

ビューア部の画面イメージを図 2-6 に示す。

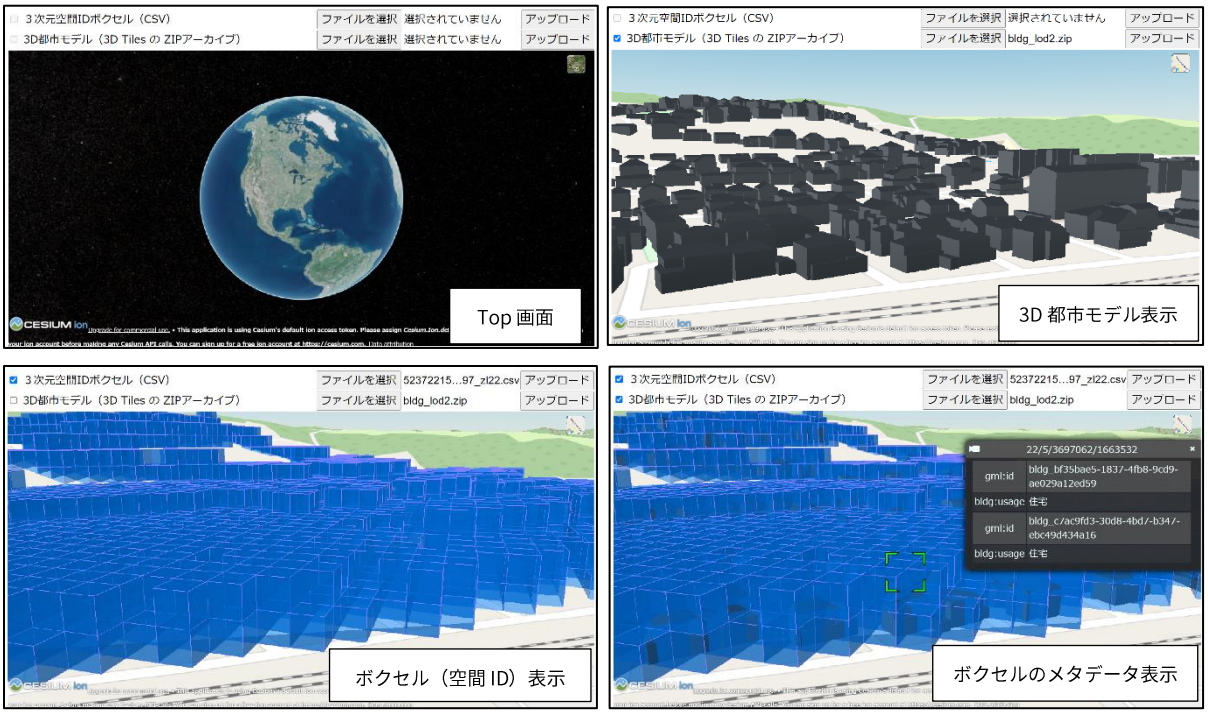


図 2-6 ビューア部の画面イメージ

3) ビューア部のソフトウェア構成

ビューア部のソフトウェア構成を図 2-7 に示す。

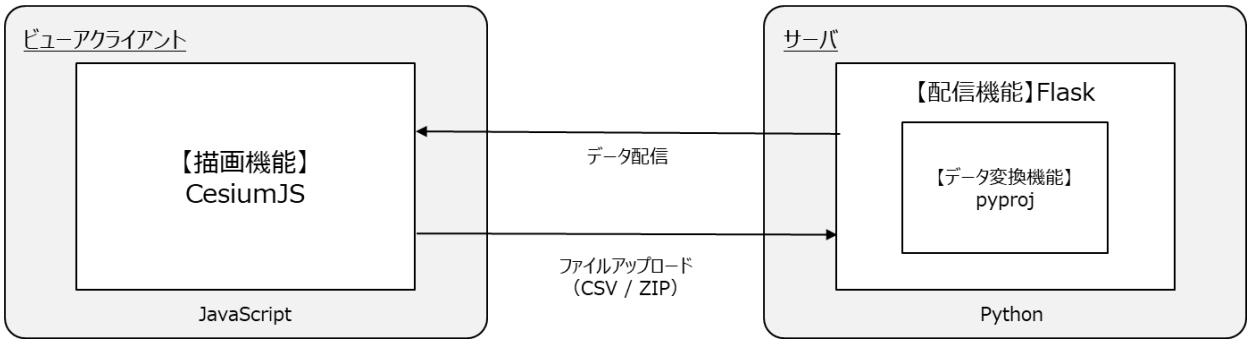


図 2-7 ビューア部のソフトウェア構成

4)　プログラム構成

ビューア部のプログラム構成を表 2-8 に示す。なお、地物 ID と空間 ID のペアリストファイルを出力するコマンド部のプログラム構成は、2.3 の 1)を参照する。

表 2-8　ビューア部のプログラム構成

プログラム構成			説明
1	サーバ	server.py	●空間 ID からボクセル形状の生成等を行うプログラム
2		cleanup.py	●一時ディレクトリの古いファイルを削除するプログラム
3	ビューア クライアント	index.js	●サーバへのファイルアップロードとサーバから配信されるデータをビューアに表示するプログラム
4		attribute.js	●空間 ID（ボクセル）のメタデータ（3D Tiles/ Mapbox Vector Tile に記録された属性）を取得するプログラム

表 2-8 に示すプログラムの関数構成を表 2-9 に示す。

表 2-9　プログラム（ビューア部）の関数構成

プログラム		関数名		説明
1	server.py	1-1	upload_tiles	●アップロードされた 3D 都市モデル（3D Tiles/ Mapbox Vector Tile）をサーバに展開し、CesiumJS で読み込むための URL をクライアントに返す ●以下の関数を内部参照 [1-2]build_tiles
		1-2	build_tiles	●3D 都市モデル（3D Tiles/ Mapbox Vector Tile）をサーバに展開 ●以下の関数を内部参照 [1-3] find_tiles [1-4] _build_tiles_2d [1-7] _build_tiles_3d
		1-3	find_tiles	●3D Tiles / Mapbox Vector Tile のファイルを検索
		1-4	_build_tiles_2d	●3D 都市モデルの Mapbox Vector Tile を GeoJSON に変換してサーバに展開 ●以下の関数を内部参照 [1-5]_arrange_feature_2d
		1-5	_arrange_feature_2d	●GeoJSON の Feature を表示用に調整 ●以下の関数を内部参照 [1-6]_arrange_geometry_2d
		1-6	_arrange_geometry_2d	●GeoJSON の Geometry を表示用に調整

プログラム		関数名		説明
		1-7	_build_tiles_3d	●3D 都市モデルの 3D Tiles をサーバに展開
		1-8	upload_csv	●アップロードされた CSV ファイルから空間 ID ボクセルを構築しクライアントに返す ●以下の関数を内部参照 [1-9]build_box
		1-9	build_box	●空間 ID から空間 ID ボクセルの Box 表現を構築 ●以下の関数を内部参照 [1-12] decode_sid [1-10]_build_box_2d [1-11] _build_box_3d
		1-10	_build_box_2d	●以下の関数を内部参照 [1-13]transform_2d
		1-11	_build_box_3d	●以下の関数を内部参照 [1-14]transform_3d [1-15]get_box_size
		1-12	decode_sid	●空間 ID をデコード
		1-13	transform_2d	●擬似メルカトル座標系から経緯度座標系（単位：ラジアン）に変換
		1-14	transform_3d	●空間 ID の代表点の座標にジオイド高を付加し CesiumJS 標準の座標系に変換
		1-15	get_box_size	●CesiumJS の Box オブジェクトのサイズを計算
2	cleanup.py	2-1	main	●一時ディレクトリの古いファイルを削除
3	index.js	3-1	uploadTiles	●3D 都市モデルの 3D Tiles/ Mapbox Vector Tile をアップロード ●以下の関数を内部参照 [3-15]setMessage [3-3]clearTiles [3-5]loadTiles
		3-2	uploadVoxels	●CSV ファイル(空間 ID ボクセル)をアップロード ●以下の関数を内部参照 [3-4]clearVoxels [2-8]loadVoxels
		3-3	clearTiles	●3D 都市モデルを破棄
		3-4	clearVoxels	●空間 ID ボクセルを破棄

プログラム		関数名		説明
		3-5	loadTiles	●3D 都市モデルの 3D Tiles/ Mapbox Vector Tile を Cesium に搭載 ●以下の関数を内部参照 [3-6]loadTiles2D [3-7]loadTiles3D
		3-6	loadTiles2D	●3D 都市モデルの Mapbox Vector Tile を Cesium に搭載 ●以下の関数を内部参照 [3-14]zoomToBounds [3-12]buildVoxelDescription2D
		3-7	loadTiles3D	●3D 都市モデルの 3D Tiles を Cesium に搭載 ●以下の関数を内部参照 [3-14] zoomToBounds [3-13] buildVoxelDescription3D
		3-8	loadVoxels	●空間 ID ボクセルを Box / Rectangle 形式で Cesium に搭載 ●以下の関数を内部参照 [3-9] loadVoxels2D [3-10] loadVoxels3D [3-14] zoomToBounds [3-12] buildVoxelDescription2D [3-13] buildVoxelDescription3D
		3-9	loadVoxels2D	●空間 ID ボクセルを Rectangle 形式で Cesium に搭載
		3-10	loadVoxels3D	●空間 ID ボクセルを Box 形式で Cesium に搭載
		3-11	setUIEventHandlers	●読み込んだデータの表示を切り替えるための UI イベントハンドラを設定
		3-12	buildVoxelDescription2D	●空間 ID（ボクセル）の表示用メタデータ（属性情報）を構築 ●以下の関数を内部参照 [4-1]getAttributes2D
		3-13	buildVoxelDescription3D	●空間 ID（ボクセル）の表示用メタデータ（属性情報）を構築 ●以下の関数を内部参照 [4-2]getAttributes3D
		3-14	zoomToBounds	●2D/ 3D の座標範囲をズーム表示

プログラム		関数名		説明
		3-15	setMessage	●各種のメッセージを出力
4	attribute.js	4-1	getAttributes2D	●Mapbox Vector Tile の Feature から表示する属性情報を取得
		4-2	getAttributes3D	●3D Tiles の Feature から表示する属性情報を取得

5)　空間 ID（ボクセル）の座標計算アルゴリズム

3D の空間 ID 文字列（{z}/{f}/{x}/{y}形式）から Cesium.Box を構築するための座標を計算するアルゴリズムは表 2-10 の通りである。

表 2-10 空間 ID（ボクセル）の座標計算アルゴリズム（3D）

手順	内容	備考
1	空間 ID の文字列を分解し z, f, x, y の各要素を整数で抽出する。	
2	ズームレベル z のボクセルの水平方向、鉛直方向の 1 辺のサイズがそれぞれ 2π*6378137 / 2 ^ z (m)、2 ^ 25 / 2 ^ z (m) になることを考慮し空間 ID(ボクセル)の座標範囲を擬似メルカトル座標系（EPSG:3857）で計算する。	
3	pyproj ライブラリを使って各座標を擬似メルカトル座標系（EPSG:3857）から EGM96 height（EPSG:5773）に変換する。その際、変換前の Z 座標を一時的に 0 に設定する。これにより水平方向は経緯度、鉛直報告は負のジオイド高が割り当てられた座標を求めることができる。	https://epsg.io/5773
4	手順 2 の Z 座標から手順 3 の Z 座標を引くことでジオイド高を加えた座標を求めることができる。	
5	pyproj ライブラリを使って手順 4 で求めた座標を経緯度座標系（EPSG:4326）から Cesium で採用されている座標系（EPSG:4978）に変換する。	https://epsg.io/4978
6	Cesium.Box は中心座標と 3 辺のサイズで指定するため頂点座標を平均して中心座標を求める。3 辺のサイズは各辺を構成する頂点を手順 5 と同様の手法で EPSG:4978 に変換した後、サイズを求める。	

2D の空間 ID 文字列（{z}/{x}/{y} 形式）から Cesium.Rectangle を構築するための座標を計算するアルゴリズムは表 2-11 の通りである。

表 2-11 空間 ID（ボクセル）の座標計算アルゴリズム（2D）

手順	内容	備考
1	空間 ID の文字列を分解し z, x, y の各要素を整数で抽出する。	
2	ズームレベル z のボクセルの水平方向が 2π*6378137 / 2 ^ z (m) になることを考慮し空間 ID(ボクセル)の座標範囲を擬似メルカトル座標系（EPSG:3857）で計算する。	
3	pyproj ライブラリを使って各座標を擬似メルカトル座標系（EPSG:3857）から 経緯度座標系（EPSG:4326）に変換する。	
4	Cesium.Rectangle は経緯度座標の最小、最大値を radian 単位で指定するため頂点座標を degree 単位から radian に変換する。	

6) 属性付与の設定

attribute.js の attribute_def_3d および attribute_def_2d 定数で空間 ID（ボクセル）に付与する 3D 都市モデル（3D Tiles/ Mapbox Vector Tile）の属性項目を設定する。

- attribute_def_3d の設定例：

```
const attribute_def_3d = [
    {name: 'attributes/gml:id', title: 'gml:id'},
    {name: '名称', title: '名称'},
    {name: 'attributes/bldg:class', title: 'bldg:class'},
    {name: 'attributes/bldg:usage/0', title: 'bldg:usage'},
    {name: 'attributes/frn:function/0', title: 'frn:function'},
    {name: 'attributes/tran:function/0', title: 'tran:function'},
    {name: 'attributes/veg:class', title: 'veg:class'},
    {name: 'attributes/wtr:function', title: 'wtr:function'},
];
```

- attribute_def_2d の設定例：

```
const attribute_def_2d = [
    {name: 'properties/gml:id', title: 'gml:id'},
    {name: 'properties/gml_id', title: 'gml:id'},
    {name: 'properties/名称', title: '名称'},
    {name: 'properties/tran:function/0', title: 'tran:function'},
    {name: 'properties/luse:class', title: 'luse:class'},
    {name: 'properties/urf:disasterType', title: 'urf:disasterType'},
    {name: 'properties/urf:function/0', title: 'urf:function'}
];
```

空間 ID に付与する 3D 都市モデルの属性の設定方法を表 2-12 に示す。

表 2-12 空間 ID に付与する属性項目の設定方法

項目名	内容	備考
name	3D 都市モデルの属性のパス	3D 地物の場合は 3D Tiles の属性、2D 地物の場合は MVT(GeoJSON)の属性の階層を定義する。項目が配列型の場合は配列のインデックスを指定する。（例：tran:function/0）
title	画面に表示する属性項目名	

なお、3D Tiles 及び MVT ファイルに記録されている 3D 都市モデルの属性の内容・構造は、FME Hub で公開されている FME ワークスペース「PLATEAU2 可視化用データ変換（https://hub.safe.com/?page=1&page_size=10&order=relevance&query=plateau）」の仕様に基づく。

7) その他の設定

PC 環境のスペックにも依存するが Cesium.Box オブジェクトや Cesium.Rectangle オブジェクトの表示数が多くなると Web ブラウザがクラッシュすることがあるため、本ツールでは Cesium.Box および Cesium.Rectangle を生成する数の上限を server.py の MAX_VOXELS に定義する。標準の設定値は 200000 である。この上限を超えた分は server.py で切り捨てた上でクライアントに表示用データを返す実装となっている。

3 ツールの利用方法

3.1 空間 ID を付与した 3D 都市モデル生成ツール

(1) 環境構築

本ツールの動作要件は次のとおり。環境構築手順を表 3-1 に示す。

- Windows / macOS / Linux
- Python 3.9

表 3-1 環境構築手順（空間 ID を付与した 3D 都市モデル生成ツール）

環境構築の手順			コマンド
1	Python 仮想環境を作成し有効化する（コマンド内の`.venv` は任意の名前に変更可能）	macOS / Linux	$ python3 -m venv .venv $ source .venv/bin/activate
		Windows	> python -m venv .venv > .venv\Scripts\Activate
2	Python 仮想環境に依存ライブラリをインストール	macOS / Linux	$ pip install -r requirements.txt
		Windows	> pip install -r requirements.txt

(2) プログラム使用方法

1) citygml2id.py

このプログラムは、CityGML から 地物 ID（gml_id） と空間 ID のペアリストを CSV 出力する。使用方法を次に示す。

```
$ python citygml2id.py [-h] [--lod {1,2,3}] [--grid-type {zfxy}] [--grid-level GRID_LEVEL] [--grid-size
  [GRID_SIZE ...]] [--grid-crs GRID_CRS] [--id [ID ...]] [--extract] [--extrude [EXTRUDE ...]]
  [--interpolate] [--merge] [--debug] input_file_or_dir output_file_or_dir
```

このプログラムの引数の説明を表 3-2 に示す。

表 3-2 citygml2id.py の引数

引数		説明
1	input_file_or_dir	CityGML のファイルのパス(*.gml) または上位ディレクトリのパス
2	output_file_or_dir	地物 ID と空間 ID のペアリストのファイルのパス(*.csv) または上位ディレクトリのパス
3	--lod	処理するジオメトリの最大 LOD(`1`, `2`, `3`)
4	--grid-type	グリッドタイプ(`zfxy`)
5	--grid-level	グリッドのズームレベル
6	--grid-size	グリッドのサイズ。x y z の順に指定。x のみ指定した場合は y z にも同じ値を適用。将来拡張用。
7	--grid-crs	グリッドの座標参照系の EPSG 番号。将来拡張用。

引数		説明
8	--id	ID フィルタ。処理するデータを絞り込む際に gml:id の値を複数指定可能。
9	--extract	空間 ID 付与済の CityGML から、空間 ID を抽出し、CSV へ出力する場合に指定。
10	--extrude	２次元データに付与する高さの最小値と最大値（単位：m）。--extract オプション指定時のみ有効。
11	--interpolate	立体（Solid）内側の空洞をボクセルで埋める場合に指定。 Solid 形状を持つ「Building（建築物）」「CityFurniture（都市設備）」「Vegetation（植生）」を空間 ID に変換する際に使用するオプション。
12	--merge	上位の空間 ID に統合（最適化）する場合に指定。
13	--debug	デバッグログ出力および一時ファイル保持を有効にする場合に指定。
14	-h	使い方を表示。

なお、出力する CSV ファイルの命名規則は次のとおりとする。

表 3-3　CSV ファイルの命名規則

ペアリストの種類	ファイル名称のルール
単一のズームレベルで出力されたペアリスト	gml ファイル名称（拡張子を除いた部分）_zl{ズームレベル}.csv （例）52372215_bldg_6697_zl20.csv
空間 ID の統合(最適化)処理がされたペアリスト	gml ファイル名称（拡張子を除いた部分）_zl{ズームレベル}_merged.csv （例）52372215_bldg_6697_zl20_merged.csv

このプログラムの使用例を以降に示す。

- [使用例 1]: ファイル単位で変換する
  - ✓ 入力：52372215_bldg_6697.gml【3D 都市モデル(CityGML)】
  - ✓ 出力：52372215_bldg_6697_zl22_merged.csv【地物 ID と空間 ID のペアリスト(CSV)】
  - ✓ グリッドタイプ：ZFXY
  - ✓ 基準（最大）ズームレベル：22
  - ✓ 空洞部の空間 ID 生成：実施
  - ✓ ズームレベル最適化：実施

```
$ python citygml2id.py D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\bldg
52372215_bldg_6697.gml D:\ 23207_toyokawa-shi_2022_citygml_1_spatialid\udx\bldg
\spatialid\52372215_bldg_6697_zl22_merged.csv --grid-type zfxy --grid-level22 --merge --
interpolate
```

- [使用例 2]: 標準製品仕様書が規定するフォルダ構成に対し一括処理をする
  - ✓ 入力：bldg フォルダ（建築物）と fld フォルダ（洪水浸水想定）
  - ✓ 出力：gml ファイルが存在するフォルダの直下に spatialid フォルダを生成し、CSV ファイルを格納
  - ✓ グリッドタイプ：ZFXY
  - ✓ 基準（最大ズームレベル）：20
  - ✓ 空洞部の空間 ID 生成：Solid 形状の建築物にのみ実施

標準製品仕様書が規定するフォルダ構成に対し一括処理する場合は、地物ごとによって適用する引数が異なる。例えば、立体（Solid）で作成される建築物は空洞部の空間 ID を生成する引数（--interpolate）を設定するが、面（Surface）で作成される水部、土地利用、都市計画決定情報等に対しては、このオプションが不要となる。このような場合では、次に示すように、各地物のフォルダに対し実行することで、それぞれの地物に応じた引数設定をすることができる。

```
$ python citygml2id.py D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\bldg
D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\bldg --grid-type zfxy --grid-level 20 –
interpolate
$ python citygml2id.py D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\fld
D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\fld --grid-type zfxy --grid-level 20
```

フォルダー括処理を実行した場合、地物 ID-空間 ID のペアリストを記録する CSV ファイルは、gml ファイルを格納するフォルダ直下に生成される「spatialid」フォルダ内に格納される（図 3-1）。

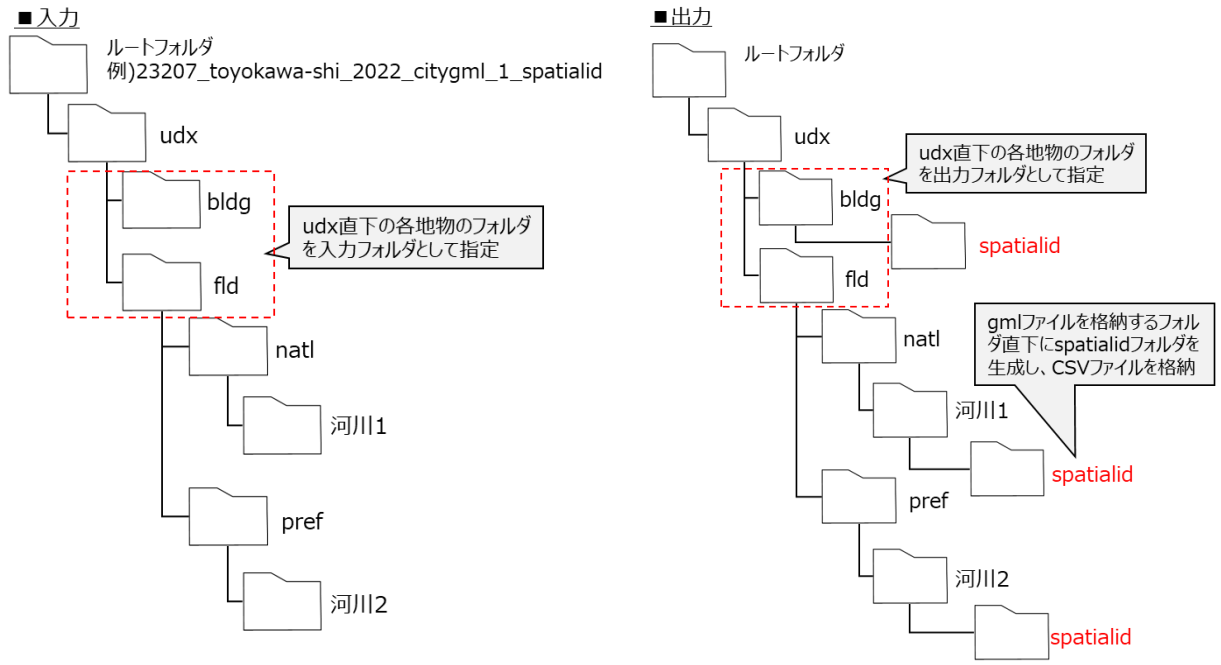


図 3-1 一括変換時のフォルダ指定

- [使用例 3]: 2 次元の空間 ID（地理院タイル(XYZ タイル)）が付与された CityGML ファイルから空間 ID を抽出し、3 次元の空間 ID（ZFXY タイル）を生成する
  - ✓ 入力：523723_luse_6668.gml
  - ✓ 出力：523723_luse_6668_zl20.csv
  - ✓ 空間 ID を生成する標高値の範囲：-10m から 100m

```
$ python citygml2id.py
D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\luse\523723_luse_6668.gml
D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\luse\\523723_luse_6668_zl20.csv
--grid-type zfxy --extract --extrude -10.0 100.0
```

図 3-2　2 次元地物から ZFXY タイルを生成

2)　id2citygml.py

このプログラムは、地物 ID と空間 ID のペアリストを記録する CSV ファイルから、CityGML ファイルを更新（空間 ID の拡張属性を新規作成又は付与済の空間 ID を更新）する。使用方法を次に示す。

```
$ python id2citygml.py [-h] citygml_file_or_dir id_file_or_dir output_file_or_dir
 [-- spatialid {embedding , reference, both}]
```

このプログラムの引数の説明を表 3-4 に示す。

表 3-4　id2citygml.py の引数

引数		説明
1	citygml_file_or_dir	CityGML のファイルのパス(*.gml) または上位ディレクトリのパス
2	id_file_or_dir	地物 ID-空間 ID ペアリストのファイルのパス(*.csv) または上位ディレクトリのパス
3	output_file_or_dir	空間 ID を付与した CityGML のファイルのパス(*.gml) または上位ディレクトリのパス
4	--spatialid	空間 ID の付与方法(`embedding`, `reference`, `both`) ・ embedding：CityGML ファイルに空間 ID を直接付与 ・ reference：地物 ID-空間 ID ペアリスト（CSV ファイル）への相対パスを記録 ⇒CityGML ファイルへの空間 ID の直接付与は行わず、外部ファイル参照のみで空間 ID と紐付けする場合に使用 ・ both：CityGML ファイルへの空間 ID 直接付与と CSV ファイルへの相対パス記録の両者を実行
5	-h	使い方を表示

このプログラムの使用例を以降に示す。

- [使用例 1]: ファイル単位で変換する
  - 入力：52372215_bldg_6697.gml
  - 地物 ID と空間 ID のペアリスト（CSV）：52372215_bldg_6697_zl20.csv
  - 出力：52372215_bldg_6697.gml
  - 空間 ID の付与方法：CityGML への直接付与と外部ファイル参照（相対パス埋め込み）の両方

```
$ python id2citygml.py
D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\bldg\52372215_bldg_6697.gml
D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\bldg\spatialid\52372215_bldg_6697_zl20.csv
D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\bldg\52372215_bldg_6697.gml
--spatialid both
```

図 3-3 に、空間 ID を付与する際の引数の設定方法の説明を示す。

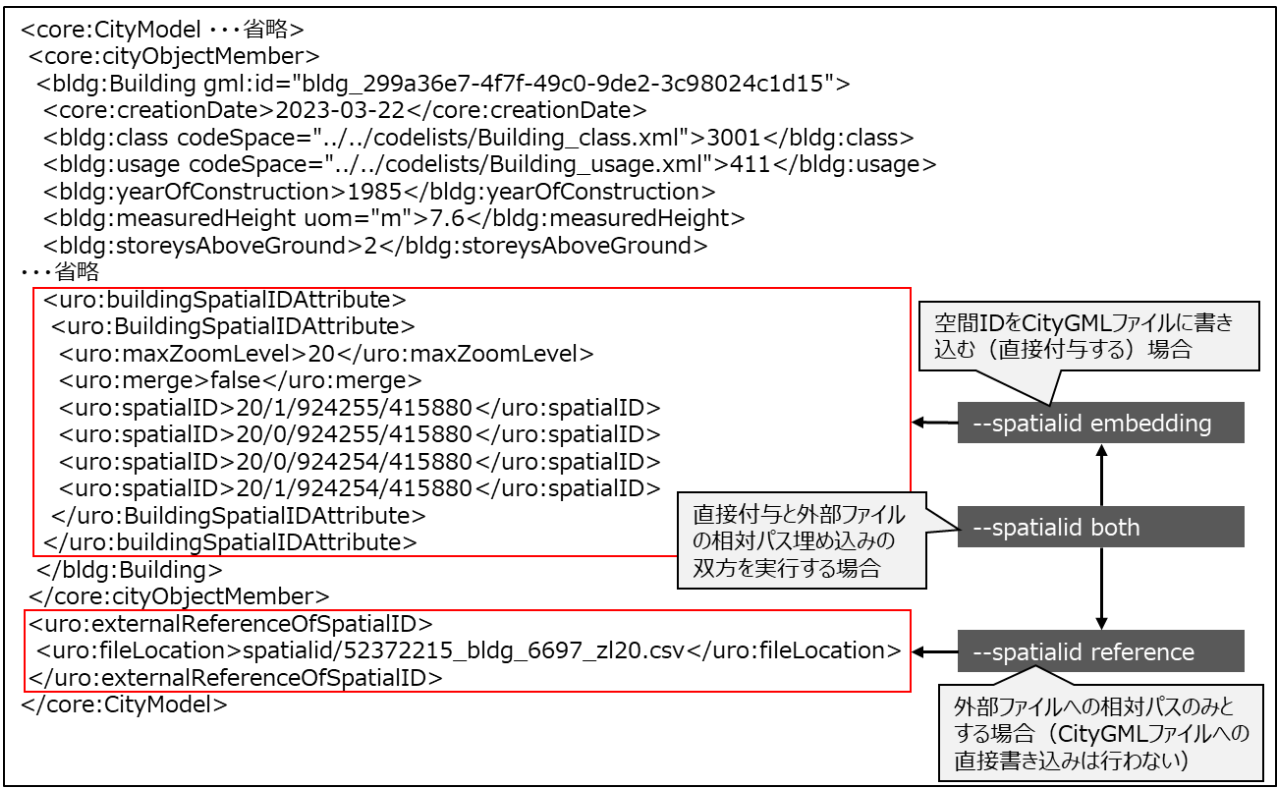


図 3-3　空間 ID 付与方法を指定する際の引数設定

- [使用例]2: 標準製品仕様書が規定するフォルダ構成に対し一括処理をする
  - 入力：bldg フォルダ（建築物）と fld フォルダ（洪水浸水想定）
  - 地物 ID と空間 ID のペアリスト（CSV）：bldg フォルダ（建築物）と fld フォルダ（洪水浸水想定）内にある spatialid フォルダ
  - 出力：bldg フォルダ（建築物）と fld フォルダ（洪水浸水想定）
  - 空間 ID 付与方法：CityGML への直接付与と外部ファイル参照（相対パス埋め込み）の両方

```
$ python id2citygml.py D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\bldg
D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\bldg D:\23207_toyokawa-
shi_2022_citygml_1_spatialid\udx\bldg --spatialid both
$ python id2citygml.py D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\fld
D:\23207_toyokawa-shi_2022_citygml_1_spatialid\udx\fld D:\23207_toyokawa-
shi_2022_citygml_1_spatialid\udx\fld --spatialid both
```

CityGML ファイルを一括更新（空間 ID の拡張属性を作成）する場合、地物 ID と空間 ID のペアリストを記録する CSV ファイルが、gml ファイルを格納するフォルダ直下の「spatialid」フォルダ内に格納されていることを前提とし（図 3-1 参照）、地物 ID-空間 ID ペアリストのファイルパスは、udx フォルダの直下にある各地物のフォルダ（例：bldg、fld 等）を指定する。

「spatialid」フォルダ内に gml ファイルに対応する複数の CSV ファイルが存在する場合は、ズームレベルが大きく、また空間 ID の最適化処理が実行されているファイルを優先し、CityGML ファイルを更新する。

3.2　空間 ID のメタデータ生成ツール（ビューア部）

(1) 環境構築

本ツールの動作要件は次のとおり。サーバの環境構築手順を表 3-5 に示す。

- サーバ
  - ✧ Windows / macOS / Linux
  - ✧ Python 3.9
- クライアント（Web ブラウザ）
  - ✧ Google Chrome
  - ✧ Microsoft Edge

表 3-5　サーバの環境構築手順（ビューア部）

環境構築の手順			コマンド
1	Python 仮想環境を作成し有効化する （コマンド内の`.venv`は任意の名前に変更可能）	macOS / Linux	$ python3 -m venv .venv $ source .venv/bin/activate
		Windows	> python -m venv .venv > .venv\Scripts\Activate
2	Python 仮想環境に依存ライブラリをインストール	macOS / Linux	$ pip install -r requirements.txt
		Windows	> pip install -r requirements.txt
3	座標変換に必要なファイルをダウンロード	macOS / Linux	$ pyproj sync --file us_nga_egm96_15.tif
		Windows	> pyproj sync --file us_nga_egm96_15.tif

開発環境で使用する場合、次の手順で開発サーバを起動する。

表 3-6　開発環境で使用する場合の手順

環境構築の手順			コマンド
1	Python 仮想環境を作成し有効化する （コマンド内の`.venv`は任意の名前に変更可能）	macOS / Linux	$ python3 -m venv .venv $ source .venv/bin/activate
		Windows	> python -m venv .venv > .venv\Scripts\Activate
2	開発サーバを起動		$ flask --app server run
3	ビューアのトップページを開く→Web ブラウザで　http://127.0.0.1:5000　を開く		

運用環境では、Apache や Nginx 等の Web サーバと mod_wsgi や uwsgi 等の WSGI 準拠のミドルウェアを組み合わせてデプロイする。

(2) ビューアの使用方法

ビューアの基本操作を図 3-4 に示す。

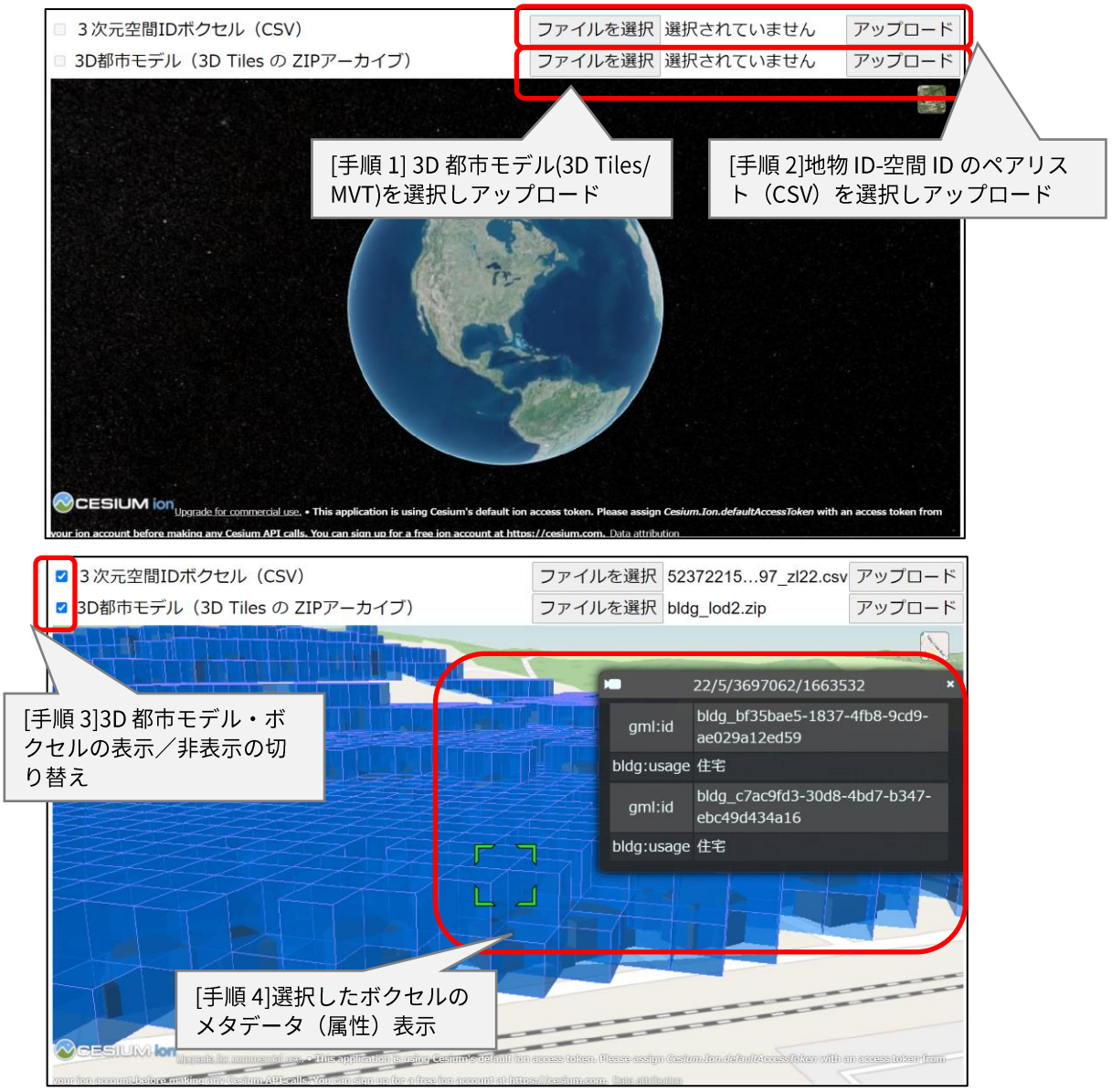


図 3-4　ビューアの基本操作

4 空間 ID 付与のための 3D 都市モデル拡張製品仕様（案）

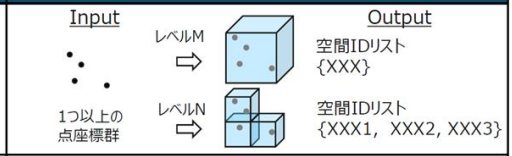
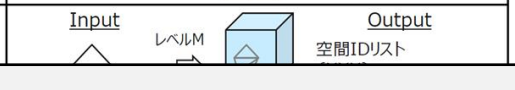
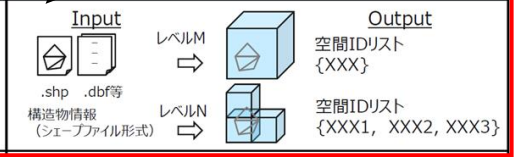
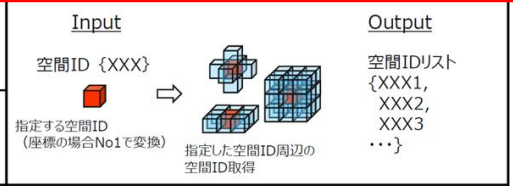
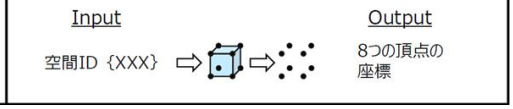
4.1 空間 ID の付与方法

「3D 都市モデル標準製品仕様書 第 2.3 版」に準拠する 3D 都市モデルに「空間 ID」を付与するために、表 4-1 に示す二つの方法を拡張仕様（案）として定義した。

表 4-1 空間 ID の付与方法

空間 ID の付与方法		説明
1	CityGML ファイルへ直接付与	●CityGML ファイル（データセット）内に記録される地物インスタンス（地物 ID）ごとに「空間 ID」を拡張属性に記録し提供 ●「空間 ID」を記録するための拡張属性を定義
2	外部ファイル（CSV 形式）に付与	●CityGML ファイルの肥大化を避ける方法として、地物 ID（gml:id）と「空間 ID」のペアリストを外部ファイルに記録し提供 ●CSV ファイルを使い「空間 ID（ボクセル）」内に存在する地物の有無を判定することができる（CityGML の解析は不要） ●「3 次元空間情報基盤」の共通機能として検討されている、領域横断的に使用される情報項目と整合（表 4-2 参照）

表 4-2 「3 次元空間情報基盤」の共通機能

No.	分類	想定する利用シーン・用途	主な入力データ	主な出力データ	実装イメージ
1	ジオメトリ⇒空間ID	点座標（複数）を空間IDに変換する	緯度・経度・高度の配列	空間IDの配列	
2	ジオメトリ⇒空間ID	閉塞範囲（穴の開いている箇所を除く）を含む空間を空間IDに変換する	緯度・経度・高度の配列	空間IDの配列	
3	ジオメトリ⇒空間ID	ウェイポイント配列を空間IDに変換する（複数の点座標を直線で繋ぎ、太さを持たせた形状）	緯度・経度・高度の配列、直径	空間IDの配列	
4	ジオメトリ⇒空間ID	3次元の構造物情報（シェープファイル形式）を空間IDに変換する	シェープファイル形式の3Dデータ	空間IDの配列	
5	周辺空間ID取得	点座標の周辺の空間を空間ID（6個または8個）に変換する	緯度・経度・高度	空間IDの配列	
6	周辺空間ID取得	空間IDを指定して周辺の空間の空間ID（6個または8個）に変換する	空間ID	空間IDの配列	
7	空間ID⇒ジオメトリ	空間IDから空間ボクセルの頂点の座標（8点）に変換する	空間ID	緯度・経度・高度の配列	

●本技術検証レポートの「空間 ID 付与のための自動生成ツール」の役割に該当する共通機能は No.4
●入力データは、「シェープファイル」ではなく「CityGML ファイル」

出典：3 次元空間情報基盤アーキテクチャ設計報告書（表を引用）
https://www.ipa.go.jp/dadc/architecture/pdf/pj_report_3dspatialinfo_doc_202208_1.pdf

「3 次元空間情報基盤」を整備する事業者は、表 4-1 に示したいずれか（又は双方）のデータを使い、3D 都市モデルのインポート機能を開発することができる。

4.2 拡張製品仕様（案）の概要

拡張製品仕様（案）には、「空間 ID」に加え、DADC との協議結果を踏まえた属性項目を定義した。拡張定義した属性項目を、表 4-3 に示す。

なお、拡張製品仕様（案）の詳細は、Project PLATEAU の GitHub に掲載する。

- 空間 ID 付与のための 3D 都市モデル拡張製品仕様書（案）
  ：https://github.com/Project-PLATEAU/PLATEAU-generator-for-spatialid/tree/main/specification

表 4-3 拡張定義した属性項目

拡張定義した属性項目		説明
1	空間 ID	●地物インスタンスが存在する空間ボクセルの識別子（空間 ID）を記録 ●2 次元の地物インスタンスについては、地理院タイル（XYZ タイル）の形式で ID を記録
2	最大ズームレベル	●3D 都市モデルのジオメトリから空間 ID を生成する際に使用した基準ズームレベル（最小のボクセルサイズ）を記録 ●この属性に記録するズームレベルを基準に、データ利用者側でより上位のズームレベルに空間ボクセルを統合することが可能 ●この属性に記録するズームレベルよりもさらに下位のズームレベル（より小さい空間ボクセル）へ分解した場合には、ジオメトリと空間 ID の存在/占有関係の正しさを保証することはできない
3	最適化処理の実施有無	●空間 ID は階層構造を持つので、上位の空間 ID を構成する 8 つのボクセルが揃っている場合、空間 ID を上位の空間 ID へ統合（最適化）しデータサイズを軽量化（空間 ID のレコード数を減らす）することができる ●3D 都市モデルのジオメトリから空間 ID を生成した際に、最適化処理を実施した結果の空間 ID が属性として記録されているかを表す（複数のズームレベルの空間 ID が属性値に混在する場合がある）

表 4-3 の空間 ID の最適化の考え方を、表 4-4 及び図 4-1 に示す。

表 4-4 拡張属性に記録する空間 ID の最適化の考え方

最適化の考え方		説明	メリット	デメリット
1	最適化無し (単一ズームレベルの空間 ID に統一)	●基準とするズームレベル（空間ボクセルの大きさ）を決め、ジオメトリが占有する空間 ID を記録	●DB への登録、検索処理等を単純化	●空間 ID のレコード数（データサイズ）が増大

最適化の考え方		説明	メリット	デメリット
2	最適化有り (複数ズームレベルの空間 ID を許容)	●基準とするズームレベルを決め、ジオメトリが占有する空間 ID を抽出し、上位の空間 ID を構成する 8 つの空間ボクセルが揃っている場合には、空間 ID を上位階層に統合	●地物のジオメトリが占有する空間 ID のレコード数を軽減 ●用途に応じ空間ボクセルを分解できる	●検索処理等が複雑化(ズームレベル間の空間ボクセルの分解・統合処理等)

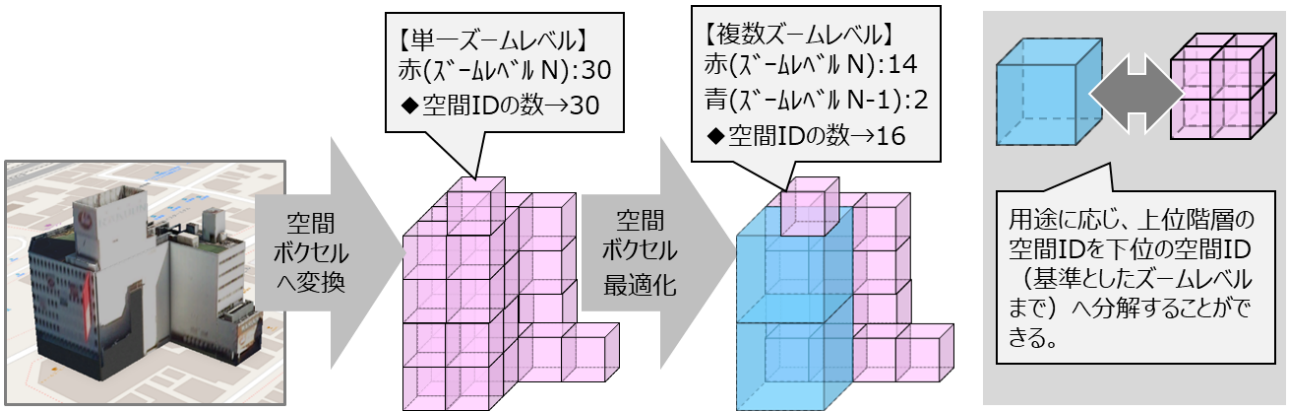


図 4-1　空間 ID の最適化の考え方

(1) CityGML ファイルの拡張仕様

空間 ID 付与のための拡張属性を、標準製品仕様書の「UrbanObject（i-UR）」パッケージと「UrbanFunction（i-UR）」に追加定義した。「Building（建築物）」の空間 ID 付与のための拡張仕様（案）を、図 4-2 に示す。同様の方法で、その他の地物にも拡張属性を定義した。

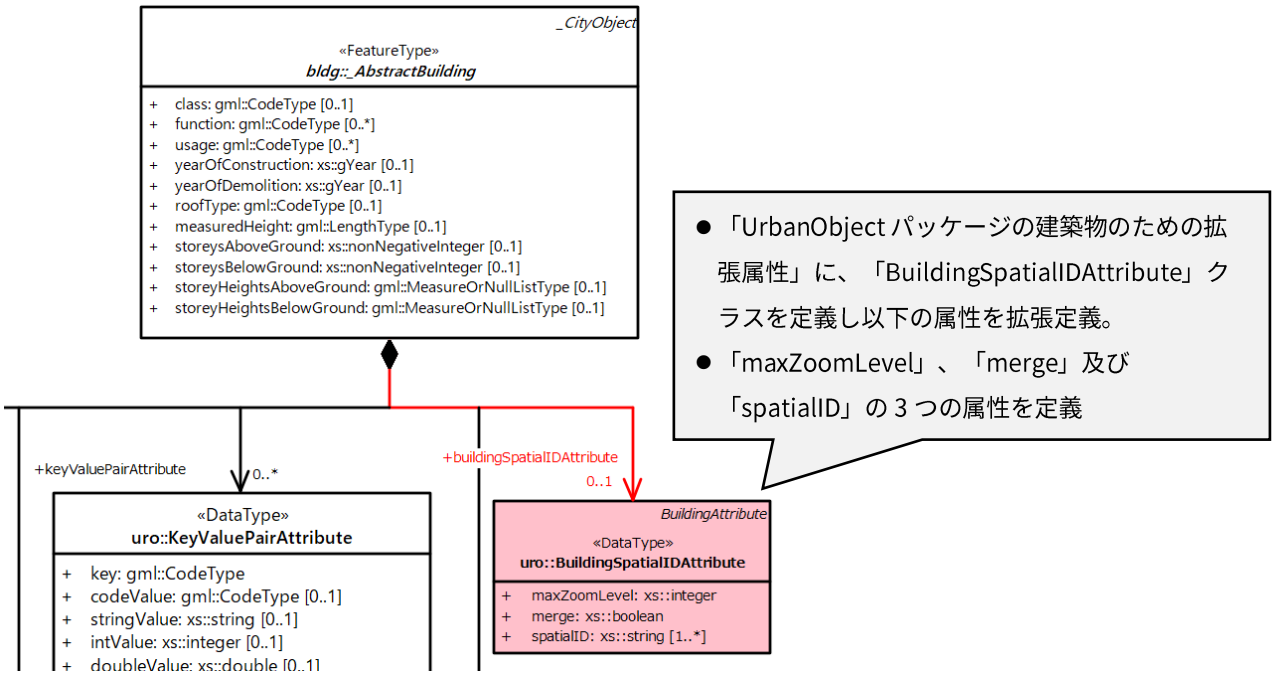


図 4-2 Building（建築物）の空間 ID 付与のための拡張仕様（案）

図 4-2 の拡張仕様（案）に基づく建築物の CityGML サンプルデータ例を図 4-3 に示す。

```
<core:cityObjectMember>
 <bldg:Building gml:id="bldg_b90c4f61-79b3-4cff-bcb7-226ab456c6c2">
  <core:creationDate>2023-03-22</core:creationDate>
  <bldg:usage codeSpace="../../codelists/Building_usage.xml">411</bldg:usage>
  ・・・
  <uro:buildingSpatialIDAttribute>
   <uro:BuildingSpatialIDAttribute>
    <uro:maxZoomLevel>22</uro:maxZoomLevel>
    <uro:merge>true</uro:merge>
    <uro:spatialID>22/7/1599930/433631</uro: spatialID >
    <uro: spatialID >22/6/1599929/433632</uro: spatialID >
    <uro: spatialID >22/7/1599929/433632</uro: spatialID >
    <uro: spatialID >22/6/1599930/433631</uro: spatialID >
    <uro: spatialID >21/3/799964/216815</uro: spatialID >
   </uro:BuildingSpatialIDAttribute>
  </uro:buildingSpatialIDAttribute>
 </bldg:Building>
</core:cityObjectMember>
```

空間 ID を記録するための拡張属性

図 4-3 空間 ID を CityGML ファイルへ直接付与した例

(2) 外部ファイル（CSV 形式）の仕様

空間 ID の外部ファイル化は、地物 ID（gml:id）と空間 ID のペアリストの情報を CSV 形式で記録する。CSV ファイルの単位は、GML ファイル（データセット）に対し、1 つの CSV ファイル内にペアリストの情報を列挙する。加えて、利用側での使いやすさを考慮し、CSV ファイルの先頭行に、ヘッダ情報として、表 4-5 に示す情報を記録する。図 4-4 に、CSV ファイルの例を示す。

表 4-5　外部ファイル（CSV 形式）に記録する情報

CSV ファイルに記録する情報			説明
1	ヘッダ情報	ヘッダ識別コード	●CSV ファイルの仕様を識別可能とし、また仕様変更が行われた際にも柔軟に対応できるよう、ヘッダ情報として記録 ●ヘッダ識別コード：PLATEAU_3D-Spatial-ID_CSV ●バージョン番号：0100（メジャーバージョン 2 桁＋マイナーバージョン 2 桁）
2		バージョン番号	
3		最大ズームレベル	●地物のジオメトリと空間 ID の関係を保証することができる最小の空間ボクセルサイズに対応したズームレベルを記録
4		最適化処理実施有無	●最適化処理の実施により、複数のズームレベルが混在する可能性があるか、0 or 1 で記録
5	ペアリストの列名	gml_id	●CityGML ファイルに記録される地物インスタンスの地物 ID（gml:id）を記録
6		spatial_id	●地物インスタンスが存在/占有する「空間 ID」を記録

```
PLATEAU_3D-Spatial-ID_CSV,0100,22,1
gml_id,spatial_id
bldg_b90c4f61-79b3-4cff-bcb7-226ab456c6c2,22/7/1599930/433631
bldg_b90c4f61-79b3-4cff-bcb7-226ab456c6c2,22/6/1599929/433632
bldg_b90c4f61-79b3-4cff-bcb7-226ab456c6c2,22/7/1599929/433632
bldg_b90c4f61-79b3-4cff-bcb7-226ab456c6c2,22/6/1599930/433631
bldg_b90c4f61-79b3-4cff-bcb7-226ab456c6c2,21/3/799964/216815
bldg_f0707e77-9065-4dd6-ac07-603846eea840,22/9/3697091/1663493
bldg_f0707e77-9065-4dd6-ac07-603846eea840,22/10/3697091/1663493
bldg_f0707e77-9065-4dd6-ac07-603846eea840,22/9/3697090/1663493
・・・
```

●先頭行：ヘッダ情報
●2 行目：列名
●3 行目以降：地物 ID と空間 ID のペアリスト

図 4-4　CSV ファイルに記録した空間 ID の例

「空間 ID」を外部ファイルに記録し提供する際に、本体の CityGML ファイルとの関連付けができるよう、データ集合のクラスである core:CityModel に拡張属性を追加した。外部ファイル参照のための拡張仕様（案）を図 4-5 に示す。CityGML ファイルから外部ファイルを参照したサンプルデータ例を図 4-6 に示す。

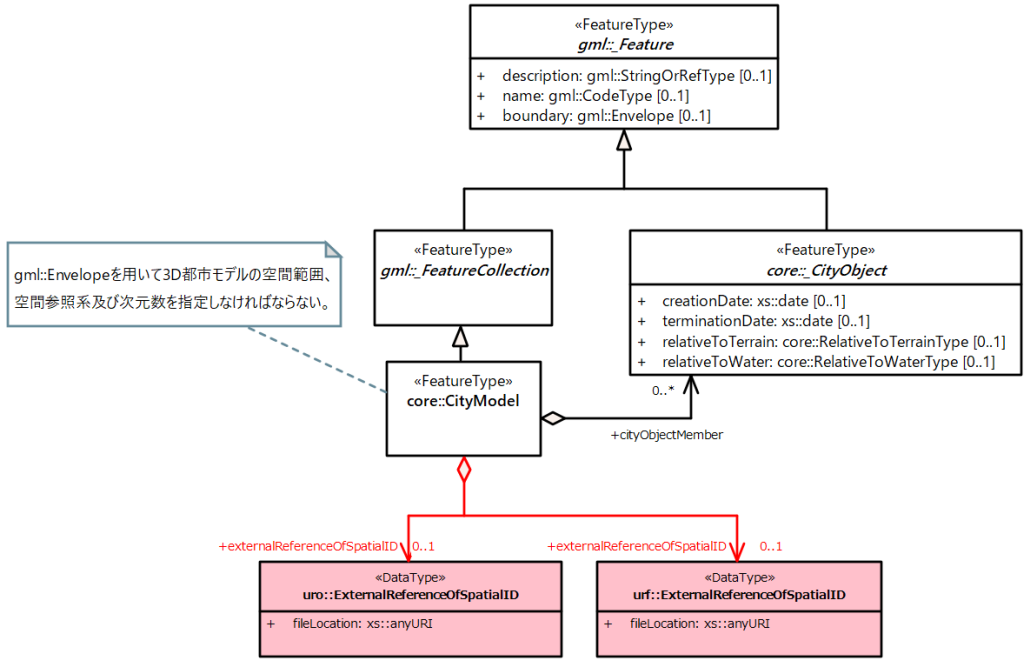


図 4-5　データ集合に関連づける外部ファイル参照のための拡張属性（案）

```
<core:CityModel xmlns:bldg="http://www.opengis.net/citygml/building/2.0" ・・・>
 <core:cityObjectMember>
  <bldg:Building gml:id="bldg_b90c4f61-79b3-4cff-bcb7-226ab456c6c2">
   ・・・
   <uro:buildingSpatialIDAttribute>
    <uro:BuildingSpatialIDAttribute>
     <uro:maxZoomLevel>22</uro:maxZoomLevel>
     <uro:merge>true</uro:merge>
     <uro:spatialID>22/7/1599930/433631</uro: spatialID >
     <uro: spatialID >22/6/1599929/433632</uro: spatialID >
     <uro: spatialID >22/7/1599929/433632</uro: spatialID >
     <uro: spatialID >22/6/1599930/433631</uro: spatialID >
     <uro: spatialID >21/3/799964/216815</uro: spatialID >
    </uro:BuildingSpatialIDAttribute>
   </uro:buildingSpatialIDAttribute>
  </bldg:Building>
 </core:cityObjectMember>
 <core:cityObjectMember>
<uro:externalReferenceOfSpatialID>
   <uro:fileLocation>spatialid/52372215_bldg_6697_zl22_merged.csv</uro:fileLocation>
 </uro:externalReferenceOfSpatialID>
</core:CityModel>
```

外部ファイル(CSV ファイル)の参照

図 4-6 CityGML ファイルから外部ファイルを参照する例

5 3D 都市モデルから生成した空間 ID の表示例

(1) 建築物

3D 都市モデルの建築物から生成した空間 ID の表示例を図 5-1 に示す。

- 対象範囲：南相馬市
- 3 次メッシュ番号：56403756

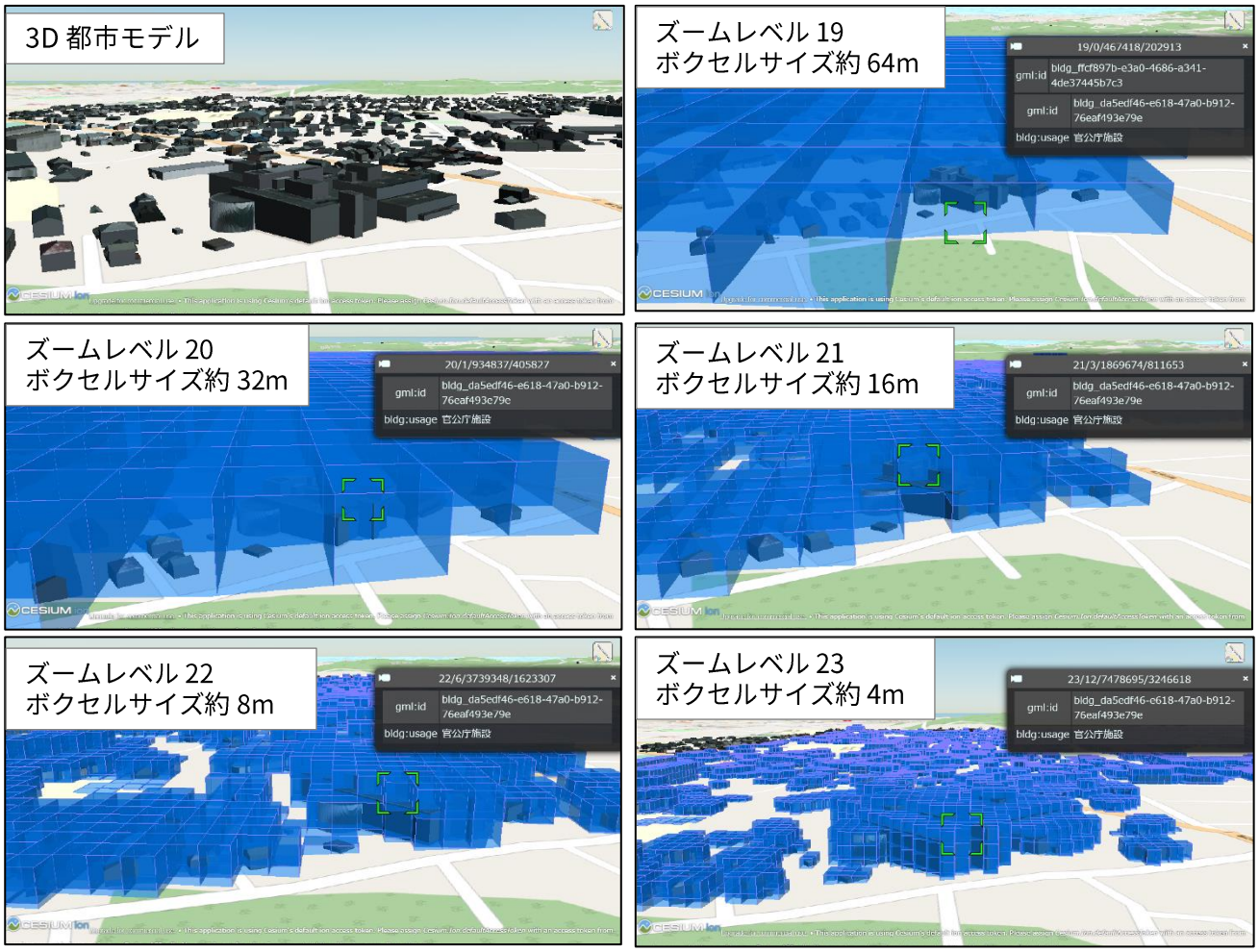
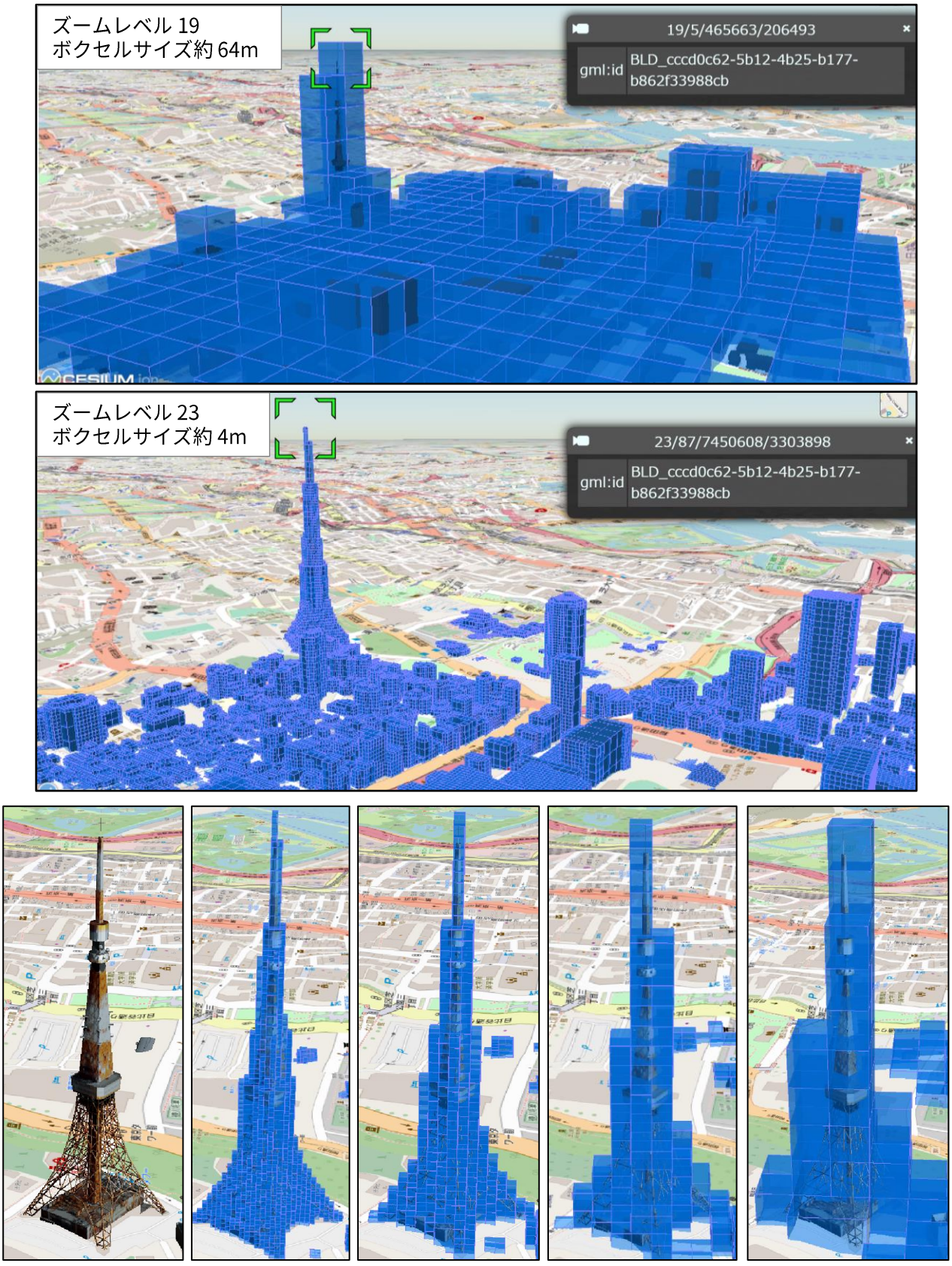


図 5-1 建築物の空間 ID 表示例

オープンデータとして公開されている東京 23 区の 3D 都市モデル（東京タワーを含む 3 次メッシュ 53393589）から生成した空間 ID の表示例を図 5-2 に示す。



3D 都市モデル　ズームレベル 23 ボクセルサイズ約 4m　ズームレベル 22 ボクセルサイズ約 8m　ズームレベル 21 ボクセルサイズ約 16m　ズームレベル 20 ボクセルサイズ約 32m

図 5-2　東京タワー周辺に生成された空間 ID

(2) 都市設備

3D 都市モデルの都市設備から生成した空間 ID の表示例を図 5-3 と図 5-4 に示す。

- 対象範囲：豊川市
- 3 次メッシュ番号：52371244

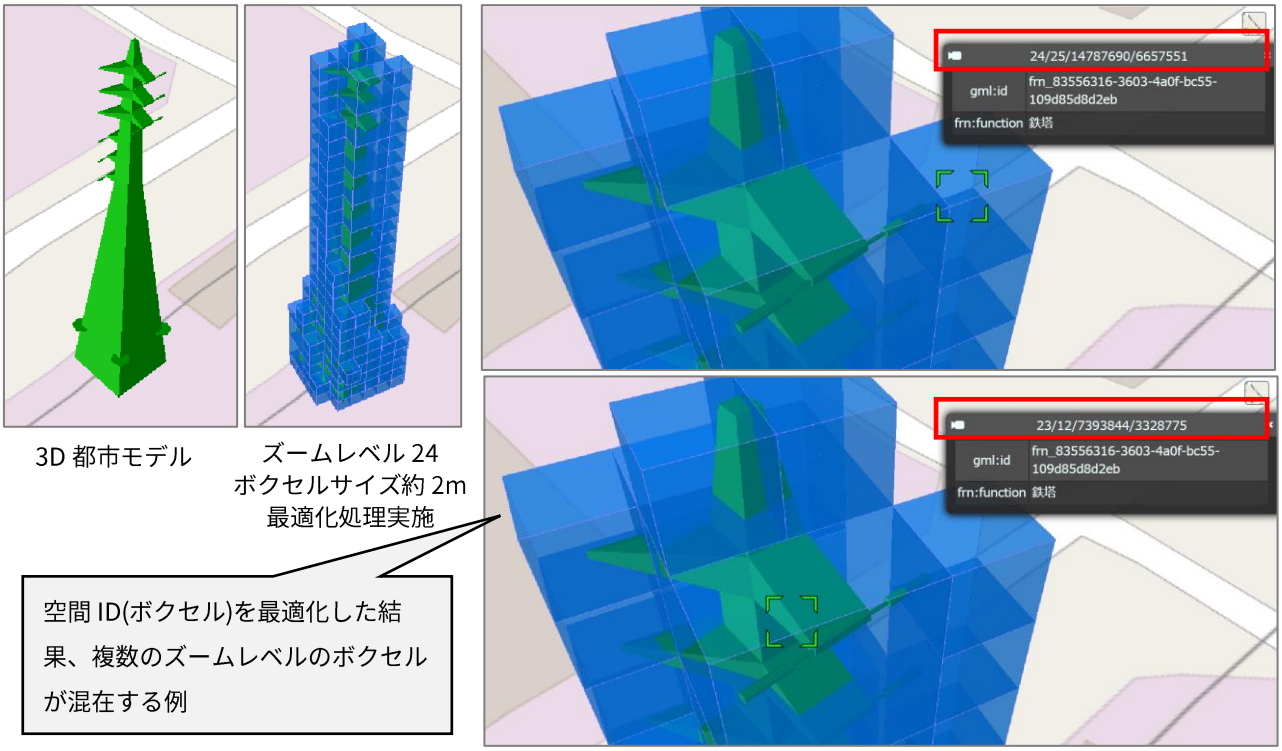


図 5-3　都市設備の空間 ID 表示例（その 1）

- 対象範囲：南相馬市
- 3 次メッシュ番号：56403756

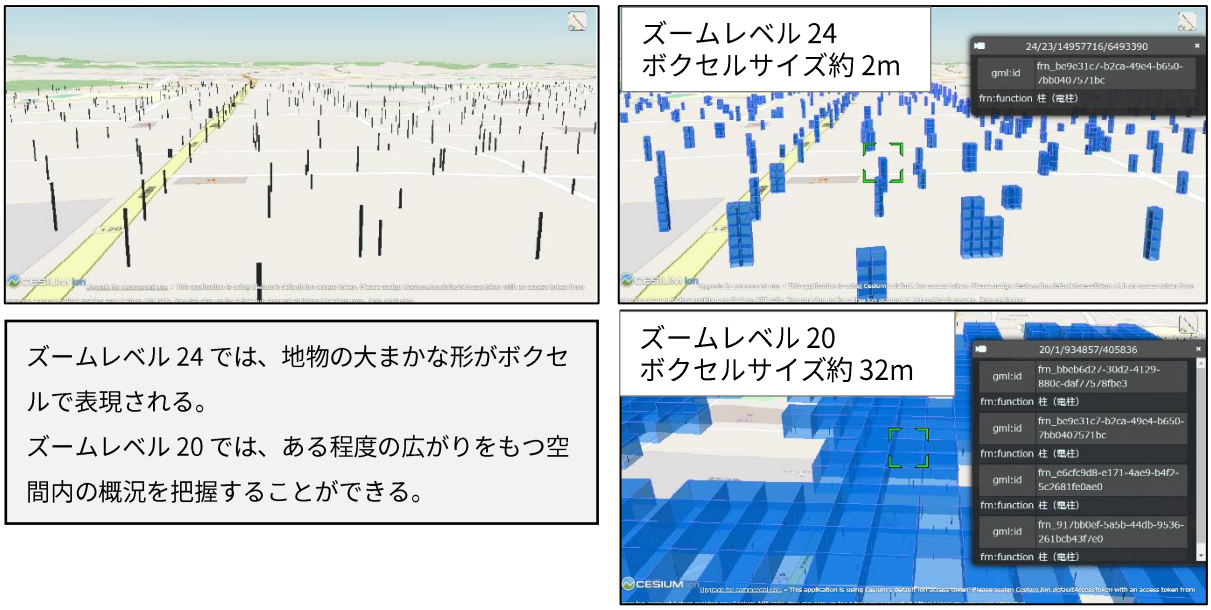


図 5-4　都市設備の空間 ID 表示例（その 2）

(3) 土地利用

3D 都市モデルの土地利用から生成した空間 ID の表示例を図 5-5 に示す。土地利用は 2 次元のデータであるため、2 次元の空間 ID（XYZ タイル）が生成される。また、XYZ タイルを元に、3 次元の空間 ID（ZFXY タイル）へ高さ方向の立ち上げを行った例も合わせて示す。

- 対象範囲：豊川市
- 2 次メッシュ番号：523723

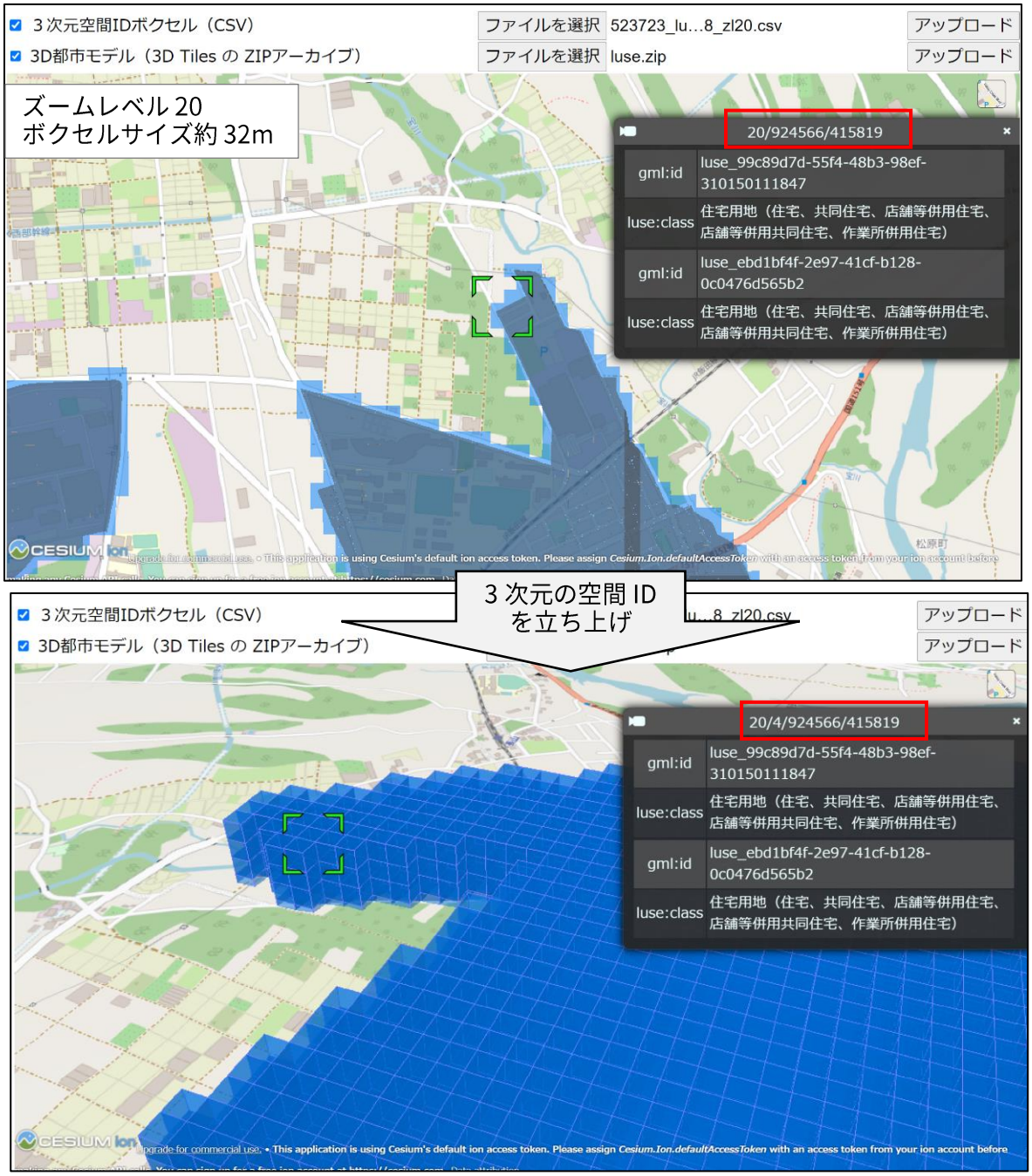


図 5-5　土地利用の空間 ID 変換例

図 5-5 は、ズームレベル 20 の ZFXY 形式の空間 ID を、標高 0m から 160m まで立ち上げた結果である。ズームレベル 20 の鉛直方向の間隔は 32m なので、{z}/{f}/{x}{y}の“f”は、“0”から“4”のボクセルが生成される。

(4) 植生

3D 都市モデルの植生から生成した空間 ID の表示例を図 5-6 に示す。

- 対象範囲：南相馬市
- 3 次メッシュ番号：56403756

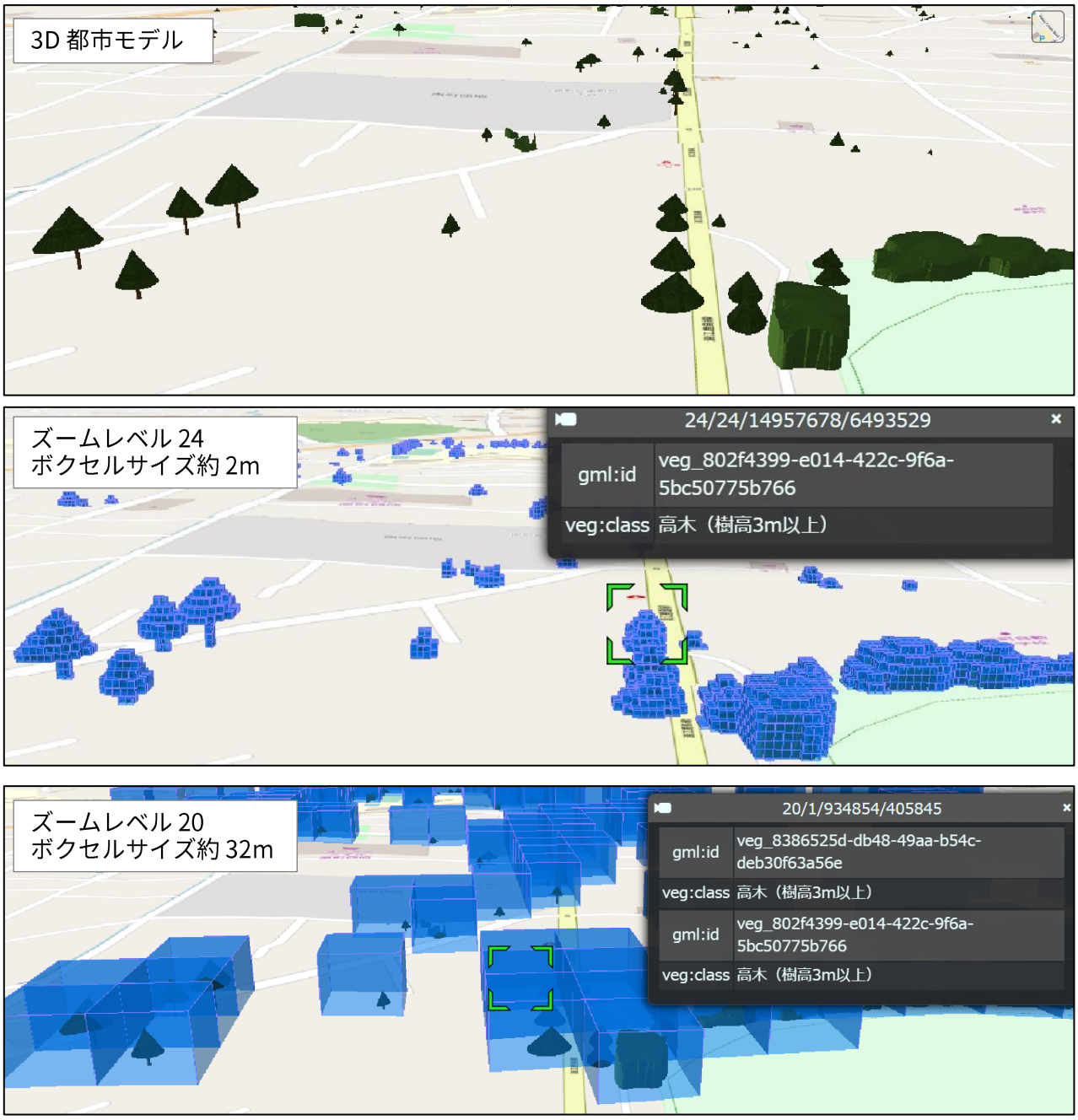


図 5-6　植生（独立樹木と植被）の空間 ID 表示例

ズームレベル 24（ボクセルサイズ 2m）の空間 ID を生成すると、3D 都市モデルの樹木の形状が再現されていることが確認できる。ズームレベル 20（ボクセルサイズ 32m）の空間 ID では、植生の大まかな分布を空間 ID（ボクセル）ごとに表すことができる。

(5) 地形

3D 都市モデルの地形から生成した空間 ID の表示例を図 5-7 に示す。

- 対象範囲：豊川市
- 2 次メッシュ番号：523722

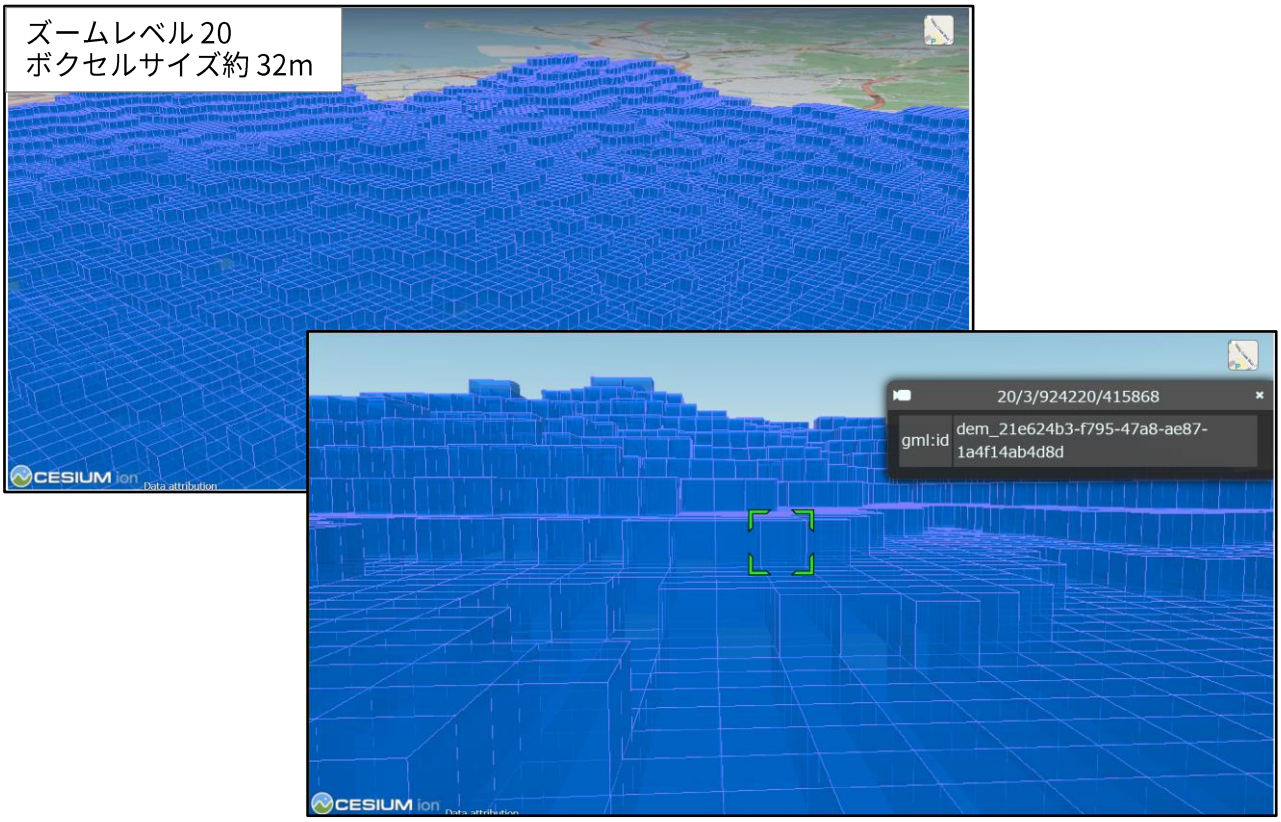


図 5-7 地形の空間 ID 表示例

(6) 土砂災害警戒区域

3D 都市モデルの土砂災害警戒区域から生成した空間 ID の表示例を図 5-8 に示す。土砂災害警戒区域は、2 次元のデータであるため、2 次元の空間 ID（XYZ タイル）に変換される。

- 対象範囲：豊川市
- 2 次メッシュ番号：523722

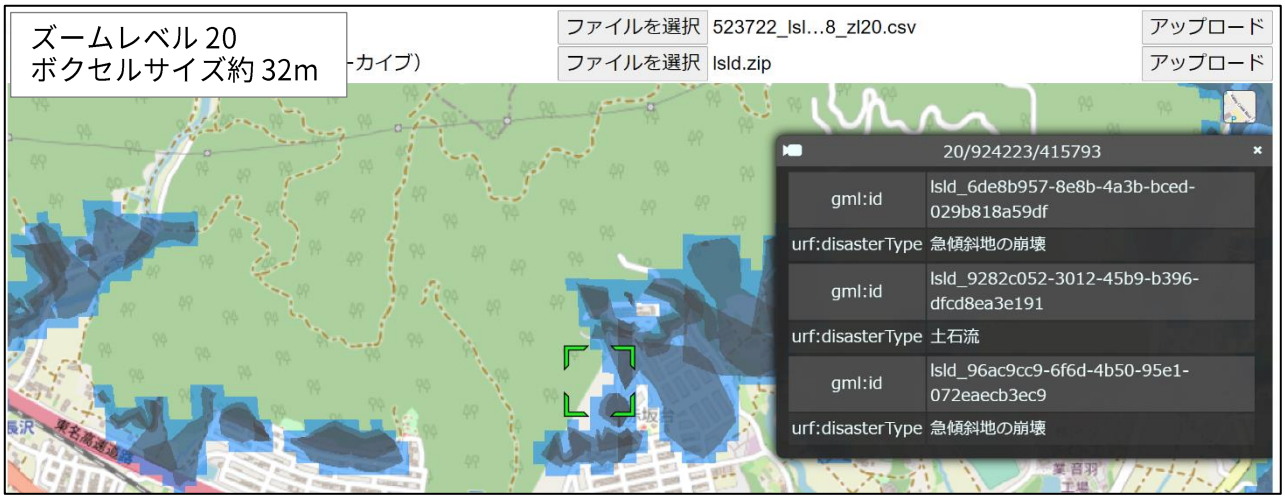


図 5-8　土砂災害警戒区域の空間 ID 表示例

(7) 道路

3D 都市モデルの道路（LOD1）に空間 ID を付与し、ビューアで表示した結果を図 5-9 に示す。道路の LOD1 は 2 次元のデータであるため、2 次元の空間 ID（XYZ タイル）に変換される。

- 対象範囲：豊川市
- 3 次メッシュ番号：52372215

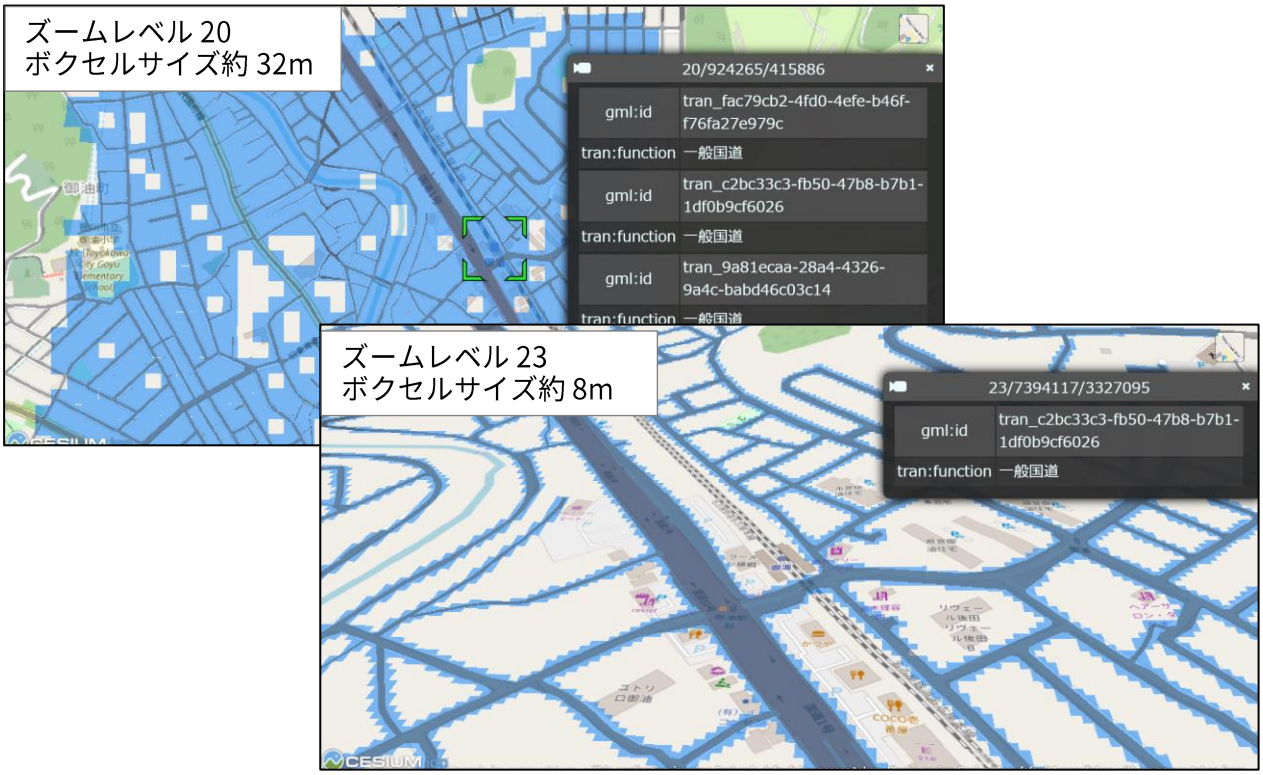


図 5-9　道路（LOD1）の空間 ID 変換例

(8) 水部（浸水想定区域）

3D 都市モデルの水部から生成した空間 ID の表示例を図 5-10 に示す。

- 対象範囲：豊川市
- 3 次メッシュ番号：52372215

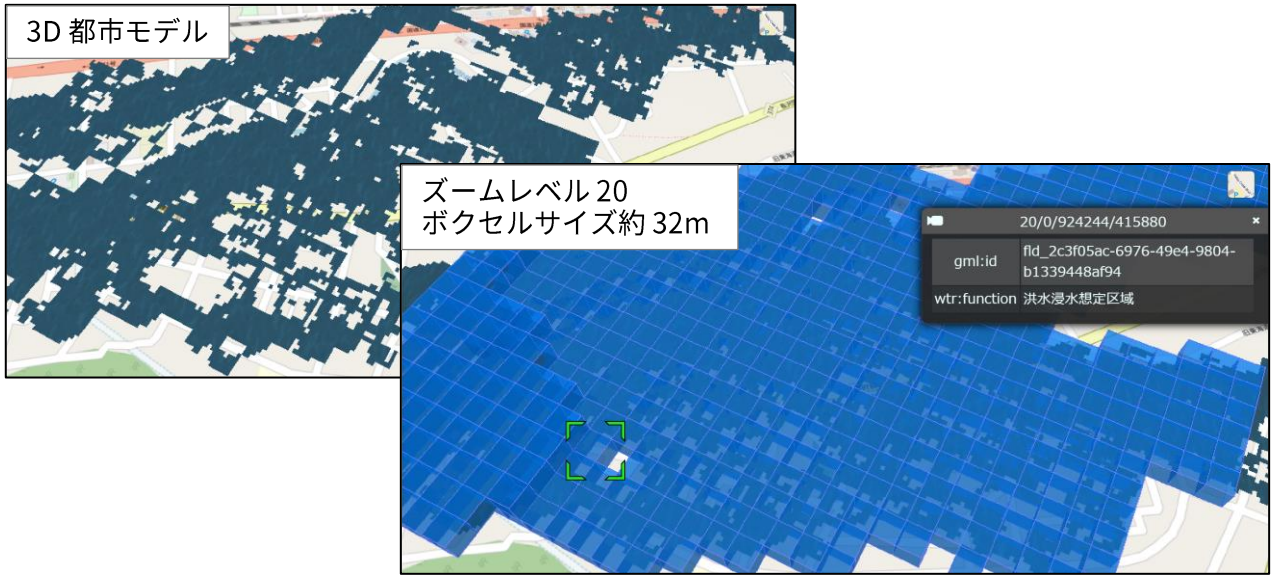


図 5-10　水部（浸水想定区域）の空間 ID 表示例

(9) 都市計画決定情報

3D 都市モデルの都市計画決定情報から生成した空間 ID の表示例を図 5-11 に示す。都市計画決定情報は 2 次元のデータであるため、2 次元の空間 ID（XYZ タイル）が生成される。XYZ タイルを元に、3 次元の空間 ID（ZFXY タイル）へ高さ方向の立ち上げを行った例も合わせて示す。

- 対象範囲：豊川市
- 3 次メッシュ番号：523712

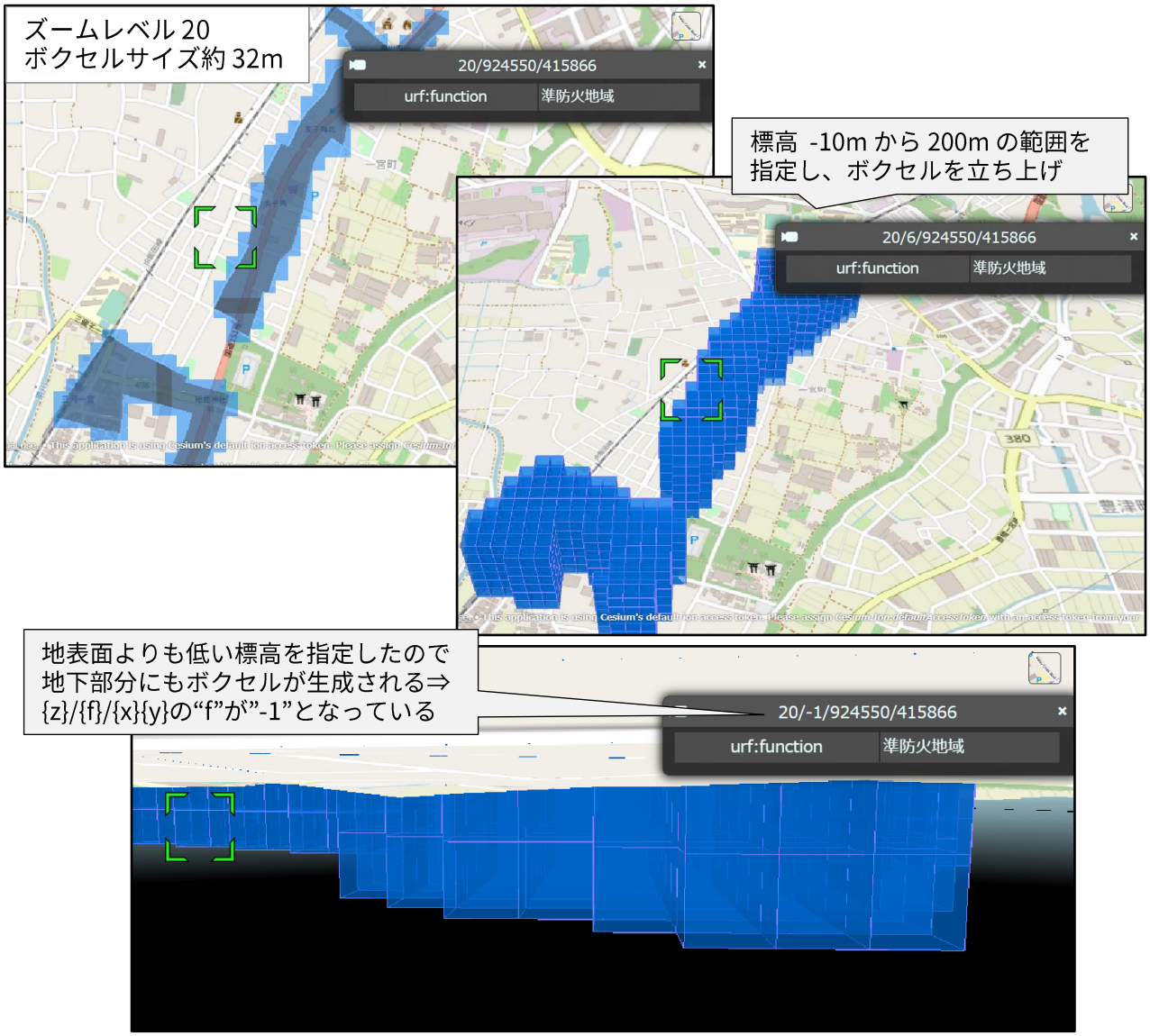


図 5-11　都市計画決定情報の空間 ID 表示

PLATEAU のための空間 ID 生成ツール　技術検証レポート

令和 5 年 3 月発行

委託者：デジタル庁

受託者：株式会社パスコ・国際航業株式会社・中日本航空株式会社共同事業体