EENVOUDIG WEBGEBRUIK VAN 3D-DATA MET INTERNATIONAAL ERKENDE STANDAARD CITYJSON

3D-representaties van onze leefomgeving zijn belangrijk in toepassingen die helpen bij de planning, de inrichting en het beheer van de openbare ruimte. Met 3D-modellen kunnen simulaties worden uitgevoerd voor bijvoorbeeld geluid, energie, luchtkwaliteit, windcomfort, zicht en wateroverlast. Om deze simulaties te bedienen met dezelfde 3D-gegevens, zijn standaarden essentieel. Voor bruikbaarheid in de praktijk moet een standaard voor 3D-data zowel specifiek als generiek zijn, eenvoudig gebruik van 3D-data mogelijk maken, en niet te grote bestanden opleveren. Dat is de motivatie geweest om CityJSON te ontwikkelen, een JSON-encoding (JavaScript Object Notation) van de OGC-standaard CityGML.

— DOOR **HUGO LEDOUX, BALÁZS DUKAI, LINDA VAN DEN BRINK, FRISO PENNINGA** EN **JANTIEN STOTER**

Al sinds 2008 werkt het Open Geospatial Consortium (OGC) aan een standaard voor 3D-modellen van de leefomgeving: CityGML. Hierbij is de aandacht vooral gericht geweest op het conceptuele model – dat verwarrend genoeg ook CityGML heet; net als de GML-encoding – met daarin de concepten, hun onderlinge relaties, hun definities, attributen enzovoort. Bij het vaststellen van de GML-encoding (Geography Markup Language) is weliswaar gekeken naar hoe het conceptuele model in XML (Extensible Markup Language) kan worden gerepresenteerd. Maar de bruikbaarheid van dit formaat heeft in de praktijk niet altijd voldoende aandacht gekregen. Daarom zijn de toepassingen voor de CityGML-encoding in de praktijk beperkt.

Op initiatief van de 3D Geoinformation onderzoeksgroep aan de TU Delft is daarom een JSON-specificatie ontwikkeld voor het conceptuele CityGML-datamodel: CityJSON (cityjson.org, zie figuur 1). CityJSON definieert hoe 3D-objecten voor gebouwen, wegen, water, bruggen en vegetatie zoals gemodelleerd in het conceptuele model CityGML, kunnen worden gerepresenteerd in JSON op verschillende detailniveaus (Levels of Detail) voor gebruik in verschillende toepassingen. De CityJSON-specificaties zijn ontwikkeld vanuit het perspectief van softwareontwikkelaars, GIS-experts en gebruikers.

Figuur 1: Logo van CityJSON

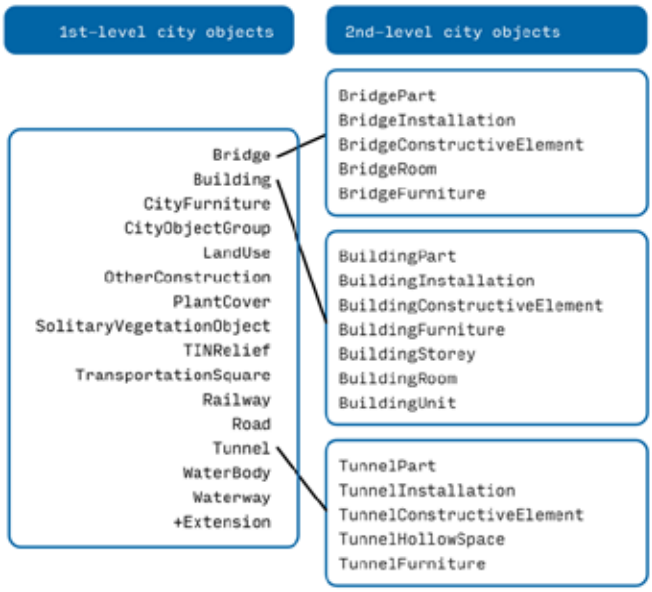
Hoe CityJSON het eenvoudig gebruik van 3D-data mogelijk maakt, blijkt uit de tientallen praktijkvoorbeelden van het gebruik van 3D BAG (zie figuur 2) die vorig jaar in CityJSON beschikbaar zijn gesteld (docs.3dbag.nl/nl/overview/media). De voorbeelden laten toepassingen zien variërend van windsimulatie, zonlicht-analyse, overstromingssimulatie en schaduwanalyse tot oplossingen ten behoeve van de energietransitie, integratie in BIM-omgevingen en stedelijke planning.

Figuur 2. 3D BAG Amsterdam in CityJSON-formaat. (Bron: www.3dbag.nl)

JSON- of GML/XML-encoding

Veel programmeertalen herkennen JSON, waar CityJSON op is gebaseerd, als een oorspronkelijk datatype dat direct kan worden gelezen en waar direct naar kan worden weggeschreven, zonder gebruik te maken van externe bibliotheken. XML, waar GML en dus CityGML op is gebaseerd, is eigenlijk geen dataformaat maar een mark-up-taal. Voordat XML bewerkt kan worden, dient de structuur eerst te worden geanalyseerd ('geparsed'). Hiervoor moet software een hiërarchie maken van de objecten in het XML-bestand, die vervolgens moet worden omgezet naar de structuur van CityGML.

Een ander voordeel van CityJSON voor softwareontwikkeling is dat het aantal manieren waarop een geometrie in 3D kan worden opgeslagen, bewust is beperkt. In CityGML zijn er 100+ manieren om dat te doen. Alleen al een simpel (2D-)vierkant kan op 25 verschillende manieren worden opgeslagen in GML, zoals wordt uitgelegd in een blog uit 2014 (zie bit.ly/GMLmadness). In 3D zijn die mogelijkheden nog talrijker. Software die CityGML ondersteunt moet alle variaties ondersteunen, en een ontwikkelaar moet ze daarom allemaal doorgronden. Daarnaast zorgt deze grote hoeveelheid aan mogelijkheden ervoor dat verschillende software op een verschillende (en dus niet gestandaardiseerde manier) met 3D-data omgaat. In CityJSON is er daarom voor gekozen het aantal modelleermogelijkheden te beperken tot alleen de meest gebruikelijke.



Figuur 4. Alle CityJSON-klassen zijn gedefinieerd als eerste of tweede orde City Objecten.

ONTWERPPRINCIPES CITYJSON

Zoals gezegd is CityJSON ontwikkeld met softwareontwikkelaars in het achterhoofd. Dit heeft geleid tot modelleerkeuzes, waarvan we er een aantal hieronder toelichten. Meer details zijn te vinden in Ledoux et al (2019).

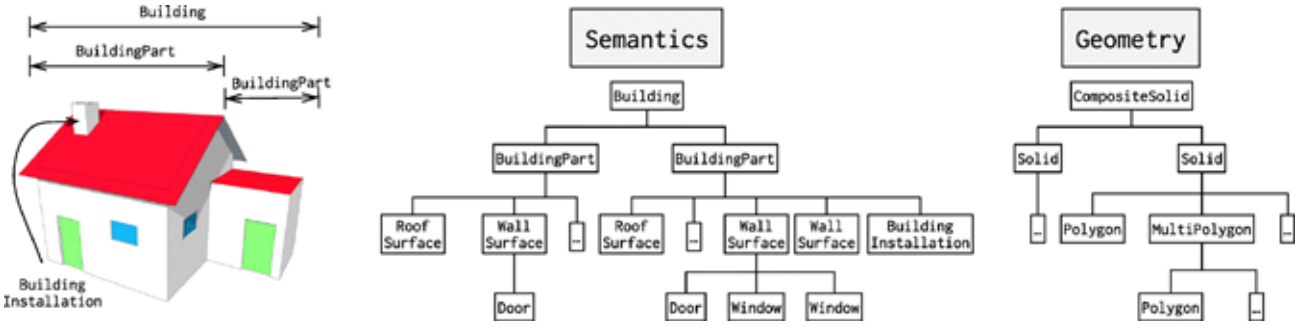
HIËRARCHIE IS VERWIJDERD

De hiërarchie van City Objects in CityGML is 'platgeslagen' in CityJSON. Figuur 3 laat als voorbeeld zien hoe in CityGML een gebouw dat uit twee delen bestaat, wordt opgedeeld in zowel geometrie als semantiek, waardoor een 'geneste' structuur ontstaat. Om deze hiërarchie per klasse te ondersteunen in software, heeft iedere klasse die ieder op een eigen manier is opgedeeld, specifieke code nodig.

In CityJSON is deze hiërarchie van overerving verwijderd. CityJSON kent alleen de eerste en tweede orde City Objecten zoals gedefinieerd in figuur 4. Elk City Object heeft vervolgens dezelfde structuur en minimaal een 'geometry' eigenschap. Attributen worden gedefinieerd in de 'attributes' eigenschap. Dit maakt het werk voor een ontwikkelaar makkelijker omdat er eenzelfde benadering is voor alle geometrieën en attributen, anders dan in de CityGML-encoding.

TOPOLOGIE

In GML, en dus de GML-encoding van CityGML, worden 3D-geometrieën opgeslagen volgens de 'Simple Features Specifications'. Dat wil zeggen dat iedere geometrische primitieve apart wordt beschreven via de coördinaten van haar vormpunten ('vertices'). Hierdoor wordt topologie niet ondersteund. CityJSON



Figuur 3. De hiërarchie voor een eenvoudig gebouw kan in CityGML behoorlijk diep worden, wat zich vertaalt in vele klassen die hiërarchisch 'genest' zijn.

daarentegen geeft eerst alle coördinaten van de vertices (voor alle objecten) in een lijst, waarna per geometrie referenties naar de positie van deze vertices in de lijst worden gegeven. Hierdoor wordt topologie expliciet opgeslagen: een robuuste oplossing voor ruimtelijke bewerkingen. Bovendien worden zo gedeelde 3D-vertices slechts eenmaal opgeslagen, wat gunstig is voor de bestandsgrootte.

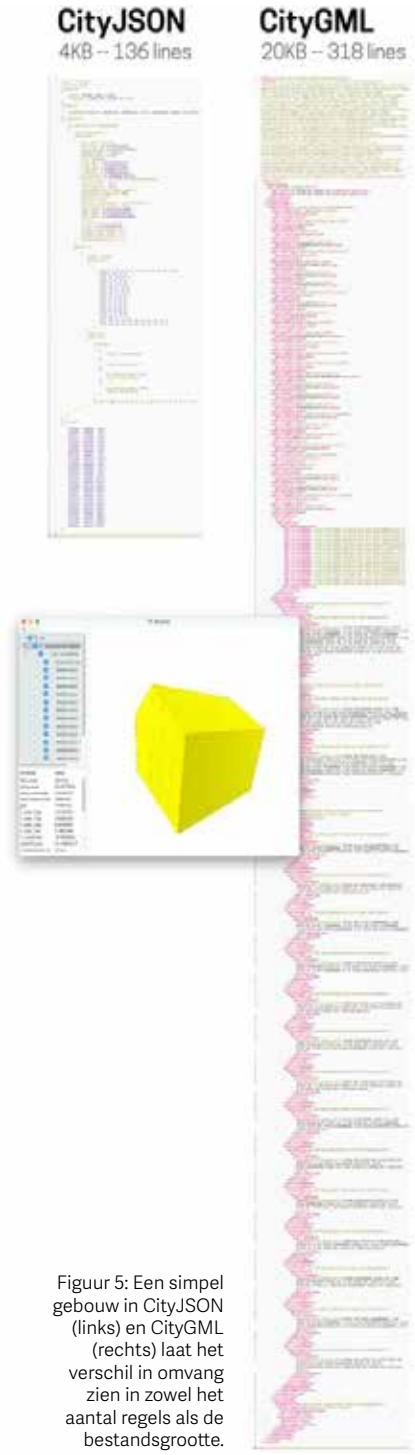
BESTANDSGROOTTE EN WEB-READINESS

Het gebruik van JSON en de modelleerkeuzes maakt de data in CityJSON veel (ongeveer zes keer) compacter dan in CityGML, zodat het mogelijk wordt om de data snel te transporteren over het internet. Figuur 5 laat bijvoorbeeld de omvang zien van een simpel gebouw, gedefinieerd in beide encodings.
Om de bestandsgrootte nog meer te verkleinen, is het mogelijk om de coördinaten van de vertices weer te geven als integers, en de schaalfactor en de translatie op te slaan die nodig is om de originele coördinaten (als floats/doubles) te verkrijgen. Dit comprimeert CityJSON-bestanden met nog eens vijf tot tien procent, en maakt de bestanden ook robuuster omdat afronding – die impact kan hebben op coördinaten – wordt voorkomen.

Volgens onderzoek (bit.ly/FHOOEAIST) is CityJSON de meest lichte 3D-standaard die ook de semantiek van objecten beschrijft. De bestanden kunnen daardoor direct worden gelezen en geschreven binnen webapplicaties, zonder dat ze eerst naar een tussenformaat worden geconverteerd. Voorbeelden van CityJSON webapplicaties die 100% Web-based zijn, zijn de drag&drop web-viewer (ninja.cityjson.org) en de validator (validator.cityjson.org).

STREAMING

Desalniettemin maakt het representeren van geometrieën door middel van referenties naar een lijst van coördinaten zoals in CityJSON, het hanteren van grote bestanden nog steeds een uitdaging. Hiervoor dient immers eerst het gehele bestand te worden ingelezen voordat de geometrie kan worden gereconstrueerd. Om streaming van CityJSON-bestanden verder te ondersteunen is recent de volgende oplossing in CityJSON geïmplementeerd. Een (groot) CityJSON-bestand wordt ontleed in de objecten die het bevat: de City Objects. Hiervan worden verschillende kleinere JSON-objecten gemaakt en deze worden opgeslagen in een JSON Lines-tekst (zie jsonlines.org). Dit formaat wordt voor dergelijke doeleinden gebruikt, zoals voor GeoJSON Text Sequences (zie bit.ly/geojsonfeaturesequences).



Figuur 5: Een simpel gebouw in CityJSON (links) en CityGML (rechts) laat het verschil in omvang zien in zowel het aantal regels als de bestandsgrootte.

Elk `CityJSONFeature` Object is een klein op zichzelf staand JSON-object, waarin één object kan worden opgeslagen, bijvoorbeeld een 'Building' met eventueel de onderliggende 'BuildingPart' en/of 'BuildingInstallation'. Hierbij worden dus alle vertices lokaal opgeslagen. De streamingsmogelijkheid van CityJSON wordt momenteel getest onder de nieuwe 'OGC API Features' (ook WFS3 genoemd), waarbij grote bestanden kunnen worden gestreamd, bevraagd en gevisualiseerd op het web.

BEPERKEN VAN MODELLEERMOGELIJKHEDEN EN VERFIJNDE LODS

CityJSON definieert dezelfde geometrische 3D-primitieven als CityGML maar negeert de geometrie-types die in de praktijk niet worden gebruikt. Point en LineString hebben bijvoorbeeld alleen hun Multi*-tegenhangers. Een enkel punt is een MultiPoint met slechts één object. Wanneer een geometrie is gedefinieerd, moet deze ook een waarde voor de LoD bevatten. Om ambiguïteit te voorkomen, wordt het gebruik van de verfijnde LoDs, zoals gedefinieerd in Biljecki (2016) aangeraden boven de vier standaard CityGML-LODs. Deze verfijning die modelleermogelijkheden nader specificeert, wordt niet in CityGML ondersteund.

VOOR SOFTWAREONTWIKKELAARS

De specificaties zijn zo kort en bondig (dat wil zeggen 'light') mogelijk geformuleerd, zodat deze gemakkelijk toegankelijk zijn voor ontwikkelaars, ook buiten het geodomein. Software en API's die CityJSON ondersteunen kunnen hierdoor snel en eenvoudig worden ontwikkeld. Support is bijvoorbeeld al beschikbaar in FME en QGIS. Ook zijn veel andere opensource-softwareoplossingen voor CityJSON ontwikkeld door ontwikkelaars over de hele wereld waaronder studenten van de MSc Geomatics TU Delft. Een lijst van software met CityJSON ondersteuning is te vinden op: www.cityjson.org/software/.

EXTENSIES

Soms is er behoefte aan meer klassen of attributen dan in CityGML worden gedefinieerd, bijvoorbeeld concepten die specifiek zijn voor de Nederlandse context. Hiervoor kent CityGML het concept van ADEs (Application Domain Extension). IMGEO is zo'n extensie (Stoter et al, 2017). Ook aan CityJSON kunnen toepassing-specifieke concepten worden toegevoegd. Dat is zo gedaan dat een CityJSON-bestand met extensies door software wordt gezien als een 'standaard' CityJSON-bestand. Er hoeft daarom geen specifieke software worden geschreven om met de toegevoegde concepten om te gaan. Dit wordt verder uitgelegd in het kader.

METADATA

Metadata is de enige toevoeging van CityJSON aan het CityGML-datamodel. Metadata zorgen ervoor dat een gebruiker – zonder data te bekijken of te downloaden – kan zien welke nauwkeurigheid en actualiteit de data hebben. Metadata maken het zoeken, vinden, downloaden en gebruik van geodata mogelijk. Zie kader voor meer details.

OGC-COMMUNITY STANDARD

Op initiatief van Geonovum is CityJSON in de zomer van 2021 voorgedragen (en geaccepteerd) als OGC-community-standaard (OGC, 2021). Dit betekent dat de OGC een standaard die buiten de OGC is ontwikkeld aanbeveelt, en die opgenomen heeft in de OGC Standards Baseline. OGC neemt het beheer van de standaard daarbij niet over.

OGC Community-standaarden hebben enerzijds als doel om geostandaarden die in de praktijk van waarde zijn tot een stabiel referentiepunt te maken, zodat overheden en andere organisaties hiernaar kunnen verwijzen. Een tweede doel is om reeds geïmplementeerde standaarden naar de OGC te brengen ter ontwikkeling van interoperabiliteit met andere OGC-standaarden. De voordracht van zo'n standaard moet worden vergezeld door bewijzen van reeds bestaande implementaties in de praktijk (dat is een voorwaarde), en moet door meerdere partijen worden onderschreven. In het geval van CityJSON waren dat, naast Geonovum en TUD, de volgende partijen: Kadaster, virtualcitySYSTEMS, National University of Singapore, Forum Virium Helsinki Oy en Ordnance Survey UK.

CITYJSON = CITYGML VERSION 3.0

In 2021 is in samenwerking met Geonovum, CityJSON aangepast op de onlangs vastgestelde nieuwe versie van CityGML. Daarmee is het de eerste (en momenteel enige) encoding van het CityGML-3.0-datamodel. Een belangrijke toevoeging aan CityGML versie 3.0 zijn de verdere opdelingen van panden (kamers, etages, appartementen, wooneenheden enzovoort); een functionaliteit waar ook in Nederland behoefte aan is.

CITYJSON IN NEDERLAND

Het Kadaster past CityJSON sinds de eerste publicatie (najaar 2020) toe voor de 3D Basisvoorziening. De 3D-informatie voor heel Nederland is in CityJSON beschikbaar en wordt bijvoorbeeld gebruikt in 3d.amsterdam.nl en 3d.utrecht.nl.
Gemeente Amsterdam bevestigt de flexibiliteit van CityJSON voor ontwikkelaars om met deze bestanden te werken. Amsterdam gebruikt Unity (WebGL) en zet de bestanden van de 3D Basisvoorziening om in binair formaat, waarbij bepaalde BGT-terreinklassen in de 'mesh' nog worden versimpeld. Amsterdam heeft ook tooling gemaakt die CityJSON-bestanden, waarin alle (bovengrondse) objecten in de stad kunnen worden gepresenteerd, inleest en hieruit datapakketjes maakt die eenvoudig in Unity kunnen worden gebruikt. Hierdoor kunnen alle objecten in de stad in Unity worden getoond zonder hiervoor extra tooling te ontwikkelen.

Extensies en Metadata

Extensies

CityJSON definieert extensies door middel van een JSON-bestand waarin de uitbreiding van de basis CityJSON wordt beschreven. Door een reeks eenvoudige, maar strikte, regels op te leggen bij het toevoegen van nieuwe concepten, wordt een CityJSON-bestand met extensies gezien als een 'standaard' CityJSON-bestand. Dit zorgt ervoor dat alle code die is geschreven om CityJSON-bestanden te verwerken, manipuleren en bekijken, zonder aanpassingen werkt op de extensies. Voorbeelden van deze regels zijn:

- De naam van een nieuw concept moet beginnen met een '+', bijvoorbeeld `+NoiseBarrier`
- Een nieuw concept moet voldoen aan de regels van CityJSON, dat wil zeggen dat het de eigenschappen 'type', 'attributes', en 'geometry' moet hebben.
- Alle geometrieën moeten als 'geometry'-eigenschap worden gedefinieerd, en kunnen niet ergens anders diep in een hiërarchie van een nieuwe eigenschap worden geplaatst.

Door deze richtlijnen zijn CityJSON-extensies weliswaar minder flexibel dan CityGML ADEs waarbij de gebruiker het datamodel op elke gewenste manier kan uitbreiden. Maar hier staat tegenover dat CityJSON-extensies kunnen worden gelezen en verwerkt door standaard CityJSON-software zonder extra aanpassingen.
Voor CityGML ADEs moet specifieke software worden geschreven, waardoor de ondersteuning van deze ADEs in de praktijk veelal beperkt is.

Metadata

In CityGML worden slechts een paar metadata-elementen ondersteund, zoals de geografische begrenzing en het CRS ('coordinate reference system'). De meeste metadata-elementen bevinden zich bovendien op het niveau van het gehele stadsmodel, en niet op het niveau van klasse of object. CityJSON heeft op basis van ISO 19115 (de ISO-standaard voor metadata van geografische informatie) een aantal metadata-kenmerken toegevoegd, en bevat aanvullende elementen die specifiek relevant zijn voor 3D-data, zoals de aanwezige detailniveaus, extensies en hun metadata.

Binnen Totaaldriedimensionaal (T3D, zie De Haan, 2021) zijn een BIM-convertor en een SketchUp-convertor ontwikkeld die de bestanden ook omzet naar CityJSON. Daarnaast is er een simpele tekentool binnen de Unity Viewer ontwikkeld waarmee een gebruiker zelf een uitbouw kan intekenen. Zodra deze gereed is, zal dit in CityJSON-formaat worden opgeslagen en kan het worden opgenomen in de 3D-registratie als nieuw object. Het CityJSON-formaat bleek bij uitstek geschikt voor deze ontwikkelingen.

Binnen T3D wordt in de context van de SOR (de Samenhangende Object Registratie) ook gekeken naar mogelijkheden van het CityGML-datamodel voor de uitbreiding in 3D. Opdeling van gebouwen is daarbij een belangrijke wens, en dat is mogelijk in CityJSON. Een andere wens binnen T3D is ook mogelijk in CityJSON: een gestandaardiseerde wijze om het CityGML datamodel uit te breiden met voor Nederland specifieke concepten zoals kabels en leidingen. Ook andere toepassingen kunnen gebruikmaken van de extensiefunctionaliteit om het datamodel gestandaardiseerd uit te breiden voor specifieke toepassingen. In Delft ontwikkelen we bijvoorbeeld een CityJSON-extensie voor Energie.

De eerdergenoemde 3D BAG van de 3D geoinformation onderzoeksgroep in Delft is ook in CityJSON beschikbaar. 3D BAG is ook beschikbaar in andere formaten zoals GeoPackage en OBJ. Maar CityJSON is de enige waarbij alle informatie over een gebouw kan worden gebundeld in één bestand, waarbij de bestandsgrootte werkbaar wordt gehouden. Ongeveer 75 procent van de gedownloade files van de 3D BAG zijn CityJSON. Dat conversie naar andere formaten triviaal is, blijkt uit de ontwikkelde software-implementaties die 3D BAG-CityJSON converteren naar OBJ, glTF, b3dm, STL of IFC (zie bit.ly/3DBAGindemedia).

Voor Geonovum bewijst deze lijst met toepassingen dat de praktijk baat heeft bij een nieuwe generatie lichtere en minder geospecifieke standaarden. Door CityJSON internationale status te geven en aan te laten sluiten bij de nieuwe versie van CityGML, streeft Geonovum naar nog bredere ondersteuning in software en verdere adoptie van de CityJSON-standaard. Potentiële gebruikers worden dan niet geconfronteerd met concurrerende standaarden, maar kunnen veilig kiezen voor het compactere en eenvoudigere CityJSON.

'Light' dataformaten hebben de toekomst, zeker bij de OGC (zie ogc.org/blog/4609). Deze zijn bij uitstek geschikt voor direct gebruik in webapplicaties en voor API-ontwikkeling. CityJSON maakt het eenvoudig gebruik van 3D-data binnen deze toekomstige ontwikkelingen mogelijk.
De ontwikkeling van CityJSON is mede mogelijk gemaakt door financiering van de European Research Council (ERC) onder het 'European Union's Horizon 2020 research and innovation programme' (Grant agreement No 677312 UMnD).

HUGO LEDOUX
werkt bij 3D Geoinformation, TU Delft. h.ledoux@tudelft.nl

BALÁZS DUKAI
werkt per februari 2022 bij de start-up 3DGI. Daarvoor was hij werkzaam bij 3D Geoinformation, TU Delft. balazs.dukai@3dgi.nl

LINDA VAN DEN BRINK
werkt bij Geonovum. l.vandenbrink@geonovum.nl

FRISO PENNINGA
werkt bij Geonovum. f.penninga@geonovum.nl

JANTIEN STOTER
werkt bij 3D Geoinformation, TU Delft. Ze is ook werkzaam bij Kadaster en Geonovum. j.e.stoter@tudelft.nl

Referenties

Biljecki F, Ledoux H, Stoter J, 2016. An improved LOD specification for 3D building models. In: Computers, Environment and Urban Systems. Elsevier: 2016. p. 25–37

Ledoux, Hugo, Ken Arroyo Ohori, Kavisha Kumar, Balázs Dukai, Anna Labetski and Stelios Vitalis, 2019. CityJSON: a compact and easy-to-use encoding of the CityGML data model. Open Geospatial Data, Software and Standards 4 (4), 2019.

Stoter, Jantien, Tom Commandeur and Hugo Ledoux, 2017. 3D BGT: waarom, wat en hoe? Geo-Info 2, 2017, blz. 69–73.

OGC, 2021. CityJSON als OGC community standard, zie docs.ogc.org/cs/20-072r2/20-072r2.html

De Haan, 2021. GeoInfo, 2021, Programma Totaal Driedimensionaal door Gerlof de Haan, Geo-Info 4, 2021, blz.43-45