

Template versie 2018-1

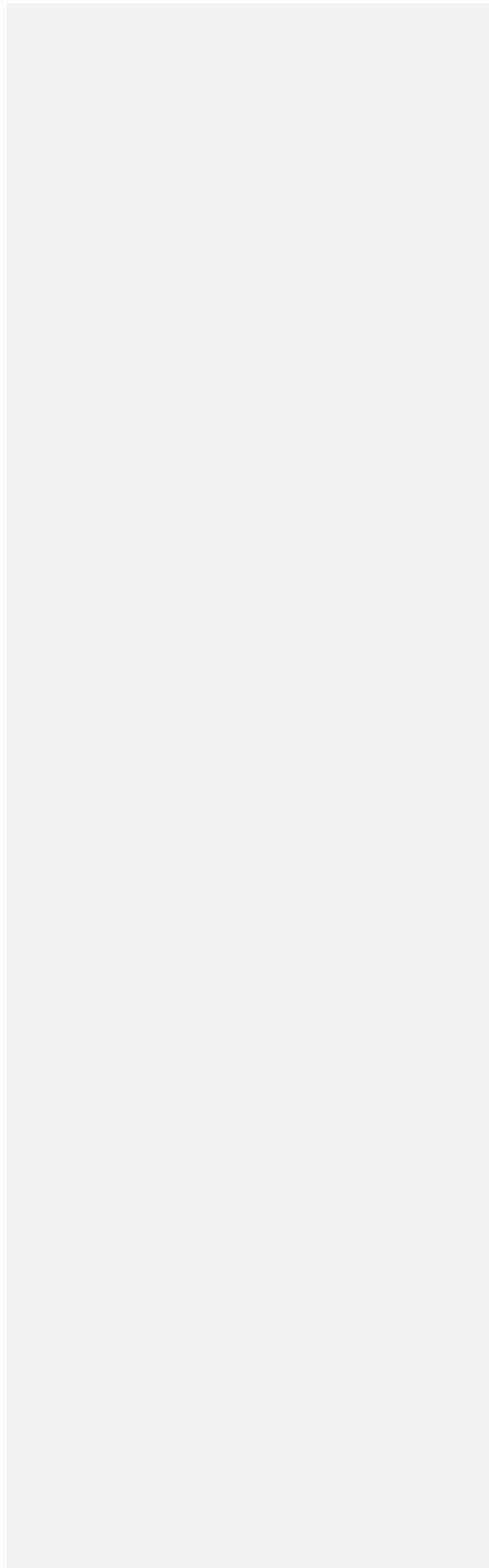
Nederlandse technische afspraak

**NTA 8035**  
(nl)

Semantische gegevensmodellering en -integratie in de gebouwde omgeving

Februari 2020

Omslag blz. 2



## Inhoud

### Inleiding 7

<b>1</b>	<b>Onderwerp en toepassingsgebied.....</b>	<b>8</b>
<b>2</b>	<b>Normatieve verwijzingen .....</b>	<b>12</b>
<b>3</b>	<b>Termen en definities .....</b>	<b>13</b>
<b>4</b>	<b>Symbolen en afkortingen.....</b>	<b>14</b>
<b>5</b>	<b>Interoperabiliteit en gegevensintegratie opties.....</b>	<b>16</b>
<b>6</b>	<b>Conceptueel Meta Model (CMM) .....</b>	<b>16</b>
<b>7</b>	<b>Taalbinding naar W3C-standaarden .....</b>	<b>30</b>
<b>8</b>	<b>Conceptueel Model (CM).....</b>	<b>51</b>
<b>9</b>	<b>Toepassing bij gegevensintegratie.....</b>	<b>62</b>
<b>10</b>	<b>Conformiteit .....</b>	<b>69</b>
<b>11</b>	<b>Bekende issues .....</b>	<b>71</b>
<b>Bijlage A (normatief) W3C-taal deelverzamelingen .....</b>		<b>74</b>
<b>Bijlage B (normatief) Resource Description Format (RDF) .....</b>		<b>79</b>
<b>Bijlage C (normatief) Turtle-formaat.....</b>		<b>88</b>
<b>Bijlage D (normatief) CMM en CM in OWL .....</b>		<b>91</b>
<b>Bijlage E (normatief) Modelleerstappen.....</b>		<b>104</b>
<b>Bijlage F (normatief) Voorbeeld van simpele Attributen en Relaties in OWL.....</b>		<b>107</b>
<b>Bijlage G (normatief) Complexe properties.....</b>		<b>112</b>
<b>Bijlage H (normatief) Voorbeeld van complexe Attributen in OWL.....</b>		<b>118</b>
<b>Bijlage I (normatief) Voorbeeld van complexe Relaties in OWL.....</b>		<b>120</b>
<b>Bijlage J (normatief) Voorbeeld van complexe Relaties voor enumeraties in OWL.....</b>		<b>122</b>

**Met opmerkingen [BH(1):** hoort dit zo (inleiding 6), ik kan hem ook niet meer updaten, graag check tom/wim

## **Voorwoord**

In april 2019 hebben de Nederlandse bouw- en infrasectoren zich ten doel gesteld een digitaal stelsel voor data op te zetten. Dit moet het huidige landschap van gefragmenteerde databronnen die niet of nauwelijks verbonden zijn tegengaan. Eilanden van data hoopt men zo te verbinden.

De recente ervaringen van afgelopen jaren met de Nederlandse COINS-standaard voor data-uitwisseling en het gebruik van ontologieën, vaak aangeduid met de term Object Type-Bibliotheken (Engels: Object Type Libraries – OTL-en) hebben ertoe geleid dat er in de Nederlandse bouw- en infrasectoren de wens is ontstaan om tot simpelere en meer standaard modelleringsafspraken te komen ten aanzien van de toepassing van de op semantiek gerichte W3C-standaarden. NTA 8035 beschrijft deze afspraken.

Deze NTA legt afspraken vast voor de toekomstvaste semantische modellering en integratie (uitwisseling of deling) van digitale gegevens (Engels: data) op basis van W3C Linked Data/Semantic Web (LD/SW)-standaarden.

Het gaat hierbij niet alleen om standaard gegevensformaten en benaderingsmethoden, maar vooral ook om gegevensstructuren (data models) die betekenis (semantiek) geven aan de gegevens.

Het zijn technische afspraken opgesteld vanuit het perspectief van partijen uit de gebouwde omgeving (gebouwen en infrastructuren) in Nederland. Met betrekking tot hun eigen ontwikkelingen kunnen partijen op deze afspraken voortborduren met waarborging op interoperabiliteit tussen de desbetreffende partijen bij de toepassing ervan.

De afspraken zijn tot stand gekomen in een kerngroep van modelleurs die de sector vertegenwoordigt en onder auspiciën van twee NEN-normcommissies heeft gewerkt:

- normcommissie 381184 'Informatie-integratie en interoperabiliteit' (spiegelgroep voor onder andere CEN/TC 442 en ISO ICDD);
- normcommissie 351225 'Regels voor informatiemodellering van de gebouwde omgeving' (betrokken bij de update van NEN 2660).

Deze NTA legt, samen met ISO ICDD, de basis voor eenduidigheid in de toepassing van semantische W3C-talen bij de toekomstvaste modellering en integratie – uitwisseling of deling – van gegevens in de gebouwde omgeving.

Deze NTA draagt hiermee bij aan een breder draagvlak en afstemming van bestaande en toekomstige ontologieën, voor zowel vraag- en aanbodzijde als implementerende softwareleveranciers.

Deze NTA is opgesteld onder verantwoordelijkheid van de normcommissie 381 184 'Informatie-integratie en interoperabiliteit', na voorbereiding door de projectgroep.

Op het ogenblik van publicatie van het normontwerp was de werkgroep van de commissie als volgt samengesteld:

M. Baggen Rijkswaterstaat

M. Böhms (editor) TNO

L. Heres Rijkswaterstaat

**NTA 8035:2020**

B. Koehorst	Rijkswaterstaat
B. Luiten	TNO
D. Oostinga	Semmtech
N. Reyngoud	Provincie Gelderland
L. van Ruijven (voorzitter)	Techniek Nederland / Croonwolter&dros
D. Spekkink	BIM Loket
S. Stolk	Semmtech
T. Hoogendijk (secretaris)	Stichting Koninklijk Nederlands Normalisatie Instituut

Deze NTA wordt ondersteund door:

**Met opmerkingen [BH(2):** geonovum nog gevraagd  
(paul/dick), TBD

M. Alamili	Sysunite
B. Bijl	Sysunite
K. Boersma	Informatiehuis Water (IHW)
T. den Braber	Moxio
S. Brouwers	CROW
S. Hendrikse	Informatiehuis Water (IHW)
N. Hoffmann	Provincie Noord-Holland
J. Honig	Bouwend Nederland
H. van der Kolk	Moxio
E. Klören	CROW
R. Kronemeijer	CROW
E. Oosterom	Stichting RIONED
R. Opgenoort	CROW
P. Willems	infraBIM
H. Winkels	BIMW
B. Witteveen	CROW
M. Vonhof	Sweco
J. Voskuil	Taxonic

**Met opmerkingen [BH(3):** ik heb nog een discussie lopen,  
final check nodig zijn naam er op mag blijven! Binnenkort  
separaat overleg met jan/sander

Schiphol er nog bij?

**Met opmerkingen [BH(4)]:** check ook bij benno

## **Inleiding**

Om interoperabiliteit bij de gegevensintegratie in de gebouwde omgeving te bereiken gaat dit document uit van NEN-ISO 16354. In hoofdstuk 5 wordt eerst dieper ingegaan op de verschillende vormen van gegevensintegratie, zoals benoemd in deze ISO-norm.

Op basis hiervan zijn de volgende twee soorten afspraken nodig:

- 1) standaardisatie van semantische modellering op metaniveau;
- 2) standaardisatie van modellering van het domein (volgens 1).

Deze NTA biedt als fundament een Conceptueel Meta Model (CMM). Het CMM bestaat uit een verzameling bouwblokken waarmee informatiemodellen kunnen worden opgebouwd, zoals Concept, Attribuut, Waarde en Relatie. Dit CMM is een RDF-gebaseerd metamodel dat onafhankelijk is van de vier talen SKOS, RDFS, OWL en SHACL, die in hoofdstuk 7 worden besproken. Kenmerkend voor RDF is het gebruik van binaire, gerichte relaties.

Hierna wordt het CMM op vier manieren naar W3C Linked Data/Semantic Web-talen/taalconstructies ‘gemapped’. Welke van de vier mappings relevant is, hangt af van de modelleerkracht die nodig is in de beoogde ‘use case’ en of er wordt gebruikgemaakt van een zogenoemde ‘open world’- of een ‘closed world’-aannname. Leidende principes zijn hier steeds ‘zo standaard mogelijk’ en ‘zo simpel mogelijk’.

In hoofdstuk 6 wordt een generieke invulling van dit CMM gegeven in de vorm van onder andere een Top Level Model. Het Concept uit het CMM wordt verder ingevuld in activiteit, gebeurtenis, fysiek object, informatieobject, toestand, ruimtelijk gebied en temporeel gebied en een aantal voorgedefinieerde relaties tussen deze concepten. Hoofdstuk 7 wordt gebruikt om dit ‘top level’ ook aan de Linked LD/SW-talen te binden.

Als er bij de mapping voor een metaconcept geen directe tegenhanger in een taal beschikbaar is, dan wordt deze gemodelleerd. Deze NTA biedt hiervoor ook een aantal standaard modelleerpatronen in hoofdstuk 8.

In hoofdstuk 9 wordt beschreven hoe we deze NTA kunnen gebruiken bij de twee hoofdvormen van gegevensintegratie: gegevensuitwisseling (hoe de NTA toe te passen in combinatie met ICDD-‘container’-) en gegevensdeling.

In hoofdstuk 10 wordt ingegaan op wat het exact betekent om volgens deze NTA data vast te leggen, uit te wisselen of te delen. In hoofdstuk 11 worden de bekende issues opgesomd, waarbij wordt aangegeven of die bij een volgende uitgave worden meegenomen.

De bijlagen beschrijven alle technische details voor de aanbevelingen die in deze NTA zijn genoemd. Het gaat bijvoorbeeld om taalbeschrijvingen, implementaties van de modelleerpatronen en voorbeeld code files.

## Semantische gegevensmodellering en -integratie in de gebouwde omgeving

### 1 Onderwerp en toepassingsgebied

De afspraken in deze NTA zijn relevant voor de gebouwde omgeving. Dit begrip omvat de Bouw (woningbouw, utiliteitsbouw en installaties) en Infra (wegen, spoorwegen, vaarwegen, watersystemen, kunstwerken als bruggen en tunnels, offshore-constructies, kabels en leidingen, enz.). Deze gegevens betreffen de gehele levenscyclus van assets: programma van eisen, ontwerp, uitvoering, beheer en onderhoud en sloop, vervanging en renovatie, hergebruik (circulair). Ook kan het gaan om alle stadia van de toeleverketen van een netwerk of portfolio van assets als wegen en gebouwen, maar ook hun systemen, bouwdelen, componenten en materialen.

Deze NTA bevat de abstracte/generieke basis voor gebouwde omgeving-specificieke gegevens. In deze NTA bevinden zich geen specifieke concepten uit de gebouwde omgeving; deze worden aan de domeinspecifieke standaarden gelaten (zoals bijvoorbeeld CB-NL in Nederland).

Gegevensintegratie kan twee verschillende vormen aannemen, die beide door deze NTA worden ondersteund:

- gegevensuitwisseling (er vindt datatransport plaats, gewoonlijk via files);
- gegevendeling (gegevens worden eenmalig vastgelegd/gepubliceerd en kunnen direct worden benaderd, gewoonlijk via internet).

Inwinning, verificatie en afleiding van gegevens spelen hierbij vaak een rol.

Gegevensstructuren (in het Engels: data models) in de vorm van vocabulaires, woordenboeken, thesauri of ontologieën spelen een sleutelrol. Deze gegevensstructuren maken de data 'semantisch', dat wil zeggen computerinterpreteerbaar (naast computerleesbaar door het juiste formaat).

Een ontologie als gegevensmodel wordt steeds meer erkend als een belangrijk instrument voor het structureren van gegevens die worden gebruikt door belanghebbenden in de bouw- en infrasectoren. Een ontologie en bijbehorende gegevensverzamelingen zijn bedoeld om bedrijfsprocessen te ondersteunen met betrekking tot alle soorten objecten/producten tijdens hun levensduur: specificatie, ontwerp, inkoop, constructie, werking of onderhoud. Een ontologie kan zowel kennis bevatten over fysieke objecten als over niet-fysieke objecten, zoals activiteiten en gebeurtenissen, met attributen, relaties, grootheden, eenheden, enz.

VOORBEELD Voorbeelden van ontologieën zijn:

- ontologieën ontwikkeld door fabrikanten om hun productgegevens te structureren;
- ontologieën ontwikkeld in het Systems Engineering (SE)-domein;
- ontologieën die door asseteigenaren zijn ontwikkeld om hun informatiebehoeften te structureren;
- ontologieën ontwikkeld op nationaal niveau en ontworpen voor hergebruik en/of koppelingsdoeleinden door/van meer gedetailleerde of meer gespecificeerde ontologieën (classificatiesystemen die verwijzen naar nationale wetgeving/normen);

## NTA 8035:2020

- ontologieën ontwikkeld op internationaal niveau en ontworpen voor hergebruik en/of koppelingsdoeleinden van nationale bibliotheken en internationale open standaarden, zoals voor BIM en GIS.

Naarmate het gebruik van ontologieën toeneemt, groeit de behoefte om te profiteren van deze ontologieën (bijvoorbeeld in ontwerpsystemen of systemen voor assetmanagement), om toegang te krijgen tot deze ontologieën en om de gestructureerde gegevens volgens die ontologieën te ontsluiten, om systeemlevenscyclusgegevens uit te wisselen en te delen tussen partijen die samenwerken in een project.

### Ambitieniveaus

Ondernemingen die gegevens willen of moeten uitwisselen/delen, hebben verschillende volwassenheidsniveaus en behoeften. Om daaraan tegemoet te komen definieert dit document drie ambitieniveaus van toenemende semantische complexiteit. Deze niveaus hebben betrekking op de verwerkbaarheid door de computer, bijvoorbeeld de mate van het automatisch kunnen interpreteren van en redeneren over de gegevens. Het hoogste niveau ondersteunt verificatie van assetgegevens tegen een gedefinieerde ontologie.

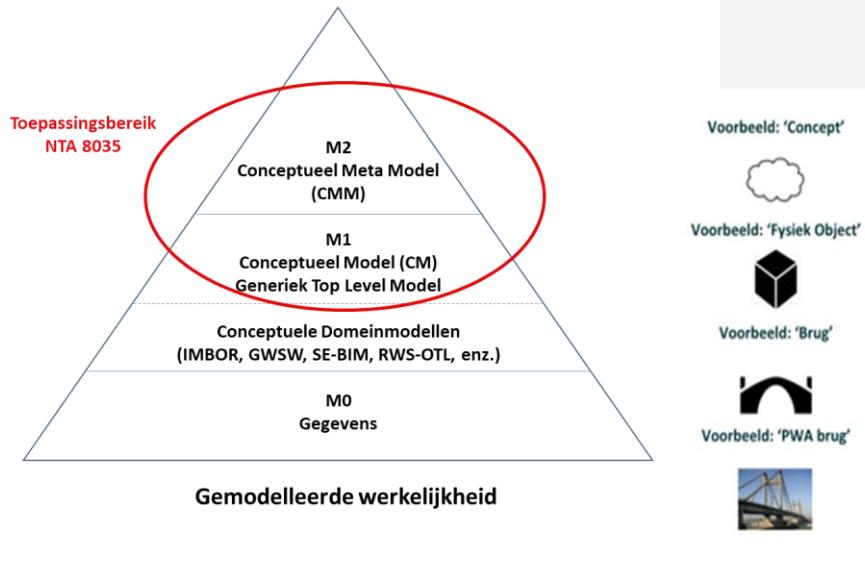
Alle drie niveaus kunnen worden toegepast om asset-gegevens te modelleren, uit te wisselen of te delen, elk met hun eigen semantische precisie.

### Soorten asset-gegevens

De reikwijdte van de asset-gegevens bestrijkt diverse 'datawerelden':

- 3D-CAD;
- Building Information Modelling (BIM);
- Geografische Informatiesystemen (GIS);
- Systems Engineering (SE);
- Productcatalogi;
- Monitoring en Control Data (sensoren/actuatoren, Internet of Things (IoT), enz.);
- Document-/Datamanagement.

Deze combinatie van werelden wordt ook wel in het algemeen aangeduid met Asset Levenscyclus Informatie Management (ALIM).



**Figuur 1 — Gelaagd modelleren**

Als het over gegevens gaat, kan er onderscheid worden gemaakt op basis van de piramide uit figuur 1:

- M0: Er zijn gegevens over aanwijsbare<sup>1</sup> dingen in de werkelijkheid (bijvoorbeeld de Prins Willem-Alexanderbrug (PWA-brug)).
- M1: Deze gegevens worden gedefinieerd door gegevensstructuren, ook wel aangeduid als conceptuele modellen. Deze kunnen generiek zijn, zoals het Conceptueel Top Level Model (CM), of domein-specifiek.
- M2: Zowel M0 als M1 bevatten soorten dingen die weer zijn vastgelegd op een hoger niveau, hier aangeduid met Conceptueel Meta Model (CMM).

De Nederlandse bouw- en infrasectoren hebben zich tot doel gesteld een digitaal stelsel op te bouwen om efficiënte en effectieve gegevensinwinning, onderlinge uitwisseling en deling tussen partners te realiseren. Het huidige datalandschap kenmerkt zich door een archipel van eilandjes van data. Er is een grote behoefte aan deze NTA, gezien de grote fragmentatie bij bestaande internationale, nationale domein- of organisatie-/project-specificheidsgegevensstructuren waardoor op dit moment een soepele integratie van gegevens sterk wordt belemmerd. Zelfs als initiatieven al uitgaan van linked data-standaarden, dan is er nog steeds een grote verscheidenheid in toepassing door de verschillende modelleerstijlen en -patronen. Deze NTA biedt hier ook de afspraken voor.

Dit zorgt ervoor dat gegevensstructuren (conceptuele modellen) gedefinieerd volgens deze NTA (op domein- of organisatie-/projectniveau), vergelijkbaarder en daardoor ook makkelijker te koppelen zijn en hierdoor in de praktijk uiteindelijk beter herbruikbaar en gecombineerd toepasbaar zijn. Kortom, een belangrijke stap om integratie van gegevens uit verschillende bronnen te bereiken.

<sup>1</sup> Of potentieel aanwijsbare dingen, bijvoorbeeld iets wat in de planning zit.

## NTA 8035:2020

Wanneer partijen volgens deze NTA werken, komt geautomatiseerde gegevensverwerking verder binnen handbereik. Hiermee wordt de kwaliteit van de uitgewisselde gegevens verbeterd in termen van consistentie en correctheid. Gegevens uit verschillende bronnen worden computerverwerkbaar en -interpreteerbaar en daarmee efficiënter en effectiever combineerbaar. Tijdrovende handarbeid om gegevens te integreren zal hierdoor verminderen, wat resulteert in lagere gegevensintegratie- en gegevenswerkingskosten en in lagere faalkosten.

Dit document biedt daarmee een oplossing om interoperabiliteit van ondernemingen, zoals gedefinieerd in NEN-ISO 11354-1, te bereiken. Hierbij wordt interoperabiliteit gedefinieerd als "het vermogen van ondernemingen en entiteiten binnen die ondernemingen om effectief te communiceren en interactief te werken". Deze NTA behandelt specifiek de interacties die plaatsvinden op gegevensniveau.

De partijen die deze NTA onderschrijven, spreken af dat bij het inrichten van de eigen gegevensmodellen die gericht zijn op onderlinge gegevensuitwisseling en -deling, wordt gebruikgemaakt van:

- het Conceptueel Meta Model (M2: Concepten, Attributen, Relaties, enz.);
- een generiek Conceptueel Top Level Model (M1), inclusief de topconcepten fysiek object, activiteit, ruimtelijk gebied, enz.;
- LD/SW-gebaseerde taalbindingen: de talen en taalconstructies voor de 'implementatie'<sup>2</sup> van het CMM en CM.

Hierbij is de keuze voor de talen en taalconstructies gebaseerd op het beoogde gebruik (term/definitie-afstemming, gegevensuitwisseling/-deling of gegevensinwinning/-verificatie/-afleiding).

Tijdens de inrichting van het Conceptueel Meta Model (M2) en het Conceptueel Top Level Model (M1) geen elementen voorkomen die specifiek zijn voor de gebouwde omgeving. Daarmee is deze NTA wellicht ook zeer goed toepasbaar buiten het perspectief van de gebouwde omgeving, dus voor andere sectoren, zoals de procesindustrie.

---

<sup>2</sup> Tussen aan- en afhalingen, want het gaat niet zozeer om software-implementatie, maar meer om 'implementatie' in een conceptuele aanbodzijde van beschikbare modelleertalen.

## 2 Normatieve verwijzingen

Naar de volgende documenten wordt in de tekst zo verwezen dat de bepalingen ervan geheel of gedeeltelijk ook voor dit document gelden. Bij gedateerde verwijzingen is alleen de aangehaalde editie van toepassing. Bij ongedateerde verwijzingen is de laatste editie van het document (met inbegrip van eventuele wijzigingsbladen en correctiebladen) waarnaar is verwezen, van toepassing.

NEN-EN-ISO 11354-1 *Advanced automation technologies and their applications - Requirements for establishing manufacturing enterprise process interoperability - Part 1: Framework for enterprise interoperability*

NEN-ISO 15704, *Enterprise modelling and architecture - Requirements for enterprise-referencing architectures and methodologies*

NEN-ISO 16354, *Guidelines for knowledge libraries and object libraries*

NEN-EN-ISO/IEC 80000, *Grootheden en eenheden - reeks*

ISO 21597-1, *Information container for linked document delivery — Exchange specification — Part 1: Container* (in voorbereiding)

ISO 21597-2, *Information container for linked document delivery — Exchange specification — Part 2: Link Types* (in voorbereiding)

*JSON-LD 1.1, A JSON-based Serialization for Linked Data*, W3C Working Draft, 18 October 2019,  
<https://www.w3.org/TR/json-ld11/>

*OWL 2 Web Ontology Language, Document Overview (Second Edition)*, W3C Recommendation, 11 December 2012, <https://www.w3.org/TR/2012/REC-owl2-overview-20121211/>

**Met opmerkingen [TH5]:** Komt nergens in voor?

*RDF 1.1 Concepts and Abstract Syntax*, W3C Recommendation, 25 February 2014,  
<https://www.w3.org/TR/rdf11-concepts/>

*RDF 1.1 Turtle*, W3C Recommendation, 25 February 2014,  
<https://www.w3.org/TR/turtle/>

*RDF Schema 1.1*, W3C Recommendation, 25 February 2014, <https://www.w3.org/TR/rdf-schema/>

**Met opmerkingen [TH6]:** Komt nergens in voor?

*SHACL (Shapes Constraint Language)*, W3C Recommendation, 20 July 2017,  
<https://www.w3.org/TR/shacl/>

*SHACL Advanced Features*, W3C Working Group Note, 08 June 2017,  
<https://www.w3.org/TR/shacl-af/>

*SKOS Simple Knowledge Organization System Reference*, W3C Recommendation, 18 August 2009,  
<https://www.w3.org/TR/skos-reference/>

*SPARQL 1.1 Overview*, 21 March 2013, W3C Recommendation,  
<https://www.w3.org/TR/sparql11-overview/>

*XML Schema Part 2: Datatypes, Second Edition*, W3C Recommendation, 28 October 2004,  
<https://www.w3.org/TR/xmlschema-2/>

### 3 Termen en definities

Voor de toepassing van dit document gelden de volgende termen en definities.

#### 3.1

##### **vocabulaire**

vocabulary

verzameling woorden die wordt gebruikt in de context van een bepaald domein (dat wil zeggen een bepaald onderwerp), een bepaalde activiteit of bij een bepaalde gelegenheid

#### 3.2

##### **woordenboek**

dictionary

bron met de woorden gebruikt in een taal (typisch in alfabetische volgorde) en hun betekenis, vaak ook met informatie over uitspraak, oorsprong en gebruik

Opmerking 1 bij de term: Een woordenboek kan ook de equivalentie woorden in een andere taal geven.

#### 3.3

##### **thesaurus**

bron met termen waar voor elke term synoniemen en gerelateerde concepten worden weergegeven

Opmerking 1 bij de term: Vaak met hiërarchische relaties als 'hoger/lager' ('broader' en 'narrower').

#### 3.4

##### **taxonomie**

taxonomy

hiërarchie van concepten waarbij de hoger/lager-rangorde wordt bepaald door specialisatie (inverse: generalisatie), ofwel superklassen/subklassen, waarbij subklassen erven van hun superklassen

Opmerking 1 bij de term: Een taxonomie heeft niet per se de vorm van een boomstructuur, het kan ook een netwerk zijn.

#### 3.5

##### **meronomie**

meronomy

hiërarchie van concepten waarbij de hoger/lager-rangorde wordt bepaald door een definiërende of typische decompositie (inverse: compositie)

Opmerking 1 bij de term: Een meronomie heeft niet per se de vorm van een boomstructuur, het kan ook een netwerk zijn.

#### 3.6

##### **gegevensmodel**

data model

gedeelde abstracte weergave van (een deel van een) domein dat voor een specifiek doel wordt gebruikt

Opmerking 1 bij de term: Voorbeelden van gegevensmodellen zijn: vocabulaires, woordenboeken, thesauri en ontologieën.

#### 3.7

##### **ontologie**

ontology

Een ontologie is een gedeelde, abstracte weergave van (een deel van) de werkelijkheid dat voor een

## **NTA 8035:2020**

bepaald doel moet worden weergegeven. Een ontologie is in wezen een samengehangend gegevensmodel bestaande uit concepten, (referentie-)instanties, waardetypen, attributen en relaties en kan ook beperkingen en afleidingen bevatten. Typisch vormen een taxonomie en/of een meronomie de ruggengraat van een ontologie

### **3.8**

#### **term**

fysieke tekenreeks die wordt gebruikt om een concept of individueel object te duiden in één of meerdere talen

### **3.9**

#### **informatieleveringsspecificatie**

Information Delivery Specification

ILS

specificatie van de content, de structuur en de dragers van de (BIM-)data die op door de opdrachtnemer gedefinieerde leveringsmomenten (Engels: data deliveries) moeten worden geleverd aan de opdrachtgever ter ondersteuning van besluitvorming in de diverse fasen van de levenscyclus van een bouwwerk (inclusief gebruik, beheer en onderhoud)

## **4 Symbolen en afkortingen**

### **4.1 Symbolen**

Symbolen komen niet voor in de NTA.

### **4.2 Afkortingen**

ALIM Asset Life-cycle Information Management

bSDD buildingSmart Data Dictionary

CC Conformance Class

CM Conceptueel Top Level Model

CMM Conceptueel Meta Model

CWA Closed World Assumption

EIR Exchange Information Requirements

GUID Globally Unique IDentifier

ICDD Information Container for linked Document Delivery

IETF Internet Engineering Task Force

ILS Informatieleveringsspecificatie

LD Linked Data

SW Semantic Web

**NTA 8035:2020**

LoC	Level of Capability
NEC	Nederlands Elektrotechnisch Comité
NTA	Nederlandse Technische Afspraak
OMG	Object Management Group
OSF	Open Software Foundation
OTL	Object Type Library
OWA	Open World Assumption
OWL	Web Ontology Language
QCR	Qualified Cardinality Restriction
QUDT	Quantities, Units, Dimensions, and Data Types
RDF	RDF Schema
SHACL	Shapes Constraint Language
SKOS	Simple Knowledge Organization System
SMLS	Semantic Modelling and Linking Standard
SSoT	Single Source of Truth
SPARQL	SPARQL Protocol And RDF Query Language
TG	Task Group
UML	Unified Modelling Language
URI	Uniform Resource Identifier
UUID	Universally Unique Identifier
W3C	World Wide Web Consortium
WG	Working Group
WWW	World Wide Web
XML	eXtensible Markup Language
XSD	XML Schema Definition language

## 5 Interoperabiliteit en gegevensintegratie opties

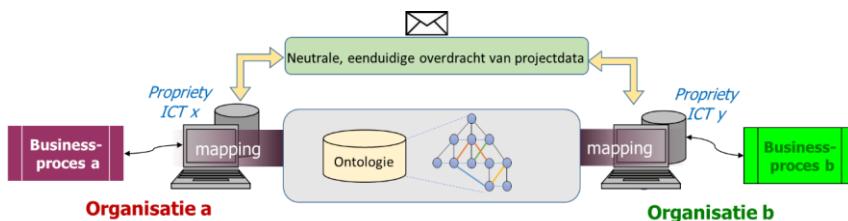
NEN-ISO 11354-1 en NEN-ISO 15704 beschrijven drie benaderingen om interoperabiliteit tussen ondernemingen te bereiken: geïntegreerd ('integrated'), verenigd ('unified') en gefedereerd ('federated'):

- Geïntegreerd: ze gebruiken precies dezelfde benadering;
- Verenigd: ze gebruiken hun eigen aanpak als speciaal geval van eenzelfde meta-afspraak;
- Gefedereerd: ze spreken niets af en bepalen 'on-the-fly' hoe zaken samenhangen.

Dit document behandelt de verenigde aanpak: een gemeenschappelijk Conceptueel Meta Model (CMM) met taalbindingen, maar ook voor een deel de geïntegreerde aanpak: een gemeenschappelijk Top Level Model.

In de verenigde aanpak wordt een gemeenschappelijke, op metaniveau gedefinieerde structuur gebruikt die wordt gehanteerd door de deelnemende partijen, in combinatie met een door de betrokken partijen gezamenlijk overeengekomen/geïntegreerde generieke 'top level'-ontologie. In deze ontologie is relevante domeinkennis geïdentificeerd en gedefinieerd en wordt op basis hiervan gedeeld. Hiermee ontstaat de mogelijkheid de semantische gelijkwaardigheid aan te geven tussen entiteiten in de gegevensbronnen van de verschillende partijen. Met behulp van de metaniveau-structuur en de top level-structuur wordt vervolgens een eenduidige integratie van gegevens tussen de verschillende partijen mogelijk.

**VOORBEELD** Figuur 2 geeft een voorbeeld in de context van (in dit geval: project-)gegevensuitwisseling. De ontologie die betekenis geeft aan de uitgewisselde gegevens, wordt bepaald door het metamodel en het generieke top level-model waarnaar de betrokken partijen vanuit hun eigen business-proces hun gegevens converteren volgens een mapping (als conversiespecificatie).



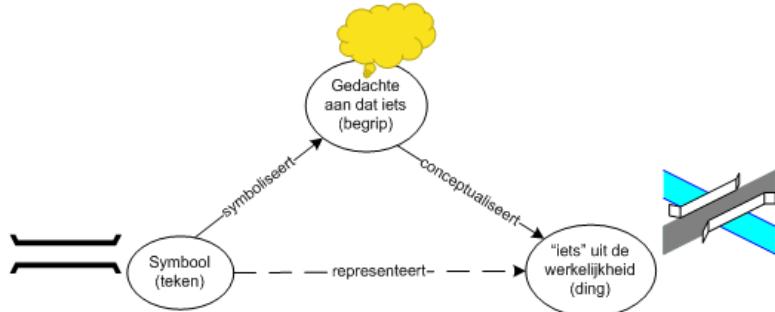
Figuur 2 — Voorbeeld bij gegevensuitwisseling van unificatie en integratie

## 6 Conceptueel Meta Model (CMM)

### 6.1 De betekenisdriehoek

In figuur 3 wordt met behulp van een driehoek de relatie gelegd tussen 'dingen', 'begrippen' en 'tekens'. De rechterbenedenhoek vertegenwoordigt 'iets uit de werkelijkheid', kortweg 'ding'. De bovenhoek staat voor 'de gedachte aan iets uit de werkelijkheid', kortweg 'begrip'. De

linkerbenedenhoek staat voor het symbool dat de gedachte symboliseert en het ‘iets’ vertegenwoordigt.<sup>3</sup><sup>4</sup>



**Figuur 3 — De betekenisdrreehoek**

**VOORBEELD** In Den Haag staat het Catshuis. Dit is wat we een ‘individueel ding’ noemen. Van dit Catshuis kunnen mensen zich een voorstelling vormen. Zo’n mentale voorstelling wordt een begrip of concept genoemd. Sommige mensen hebben het Catshuis zelf gezien. Anderen kennen het alleen van een foto. Weer anderen kennen het alleen van naam. Hun voorstellingen van het Catshuis zullen dus onderling verschillen.

Elk begrip kan met een symbool, een teken, worden verbonden. In het geval van het Catshuis is dat onder meer de eigenaam Catshuis. Deze naam kan worden uitgesproken of worden geschreven: C-a-t-s-h-u-i-s. Gebarentolkens hebben er een speciaal gebaar voor. Maar er zijn ook andere symbolen denkbaar. Bijvoorbeeld een pictogram of een tekening. Ook een digitaal 3-D-model van het Catshuis kan als een symbool worden beschouwd.

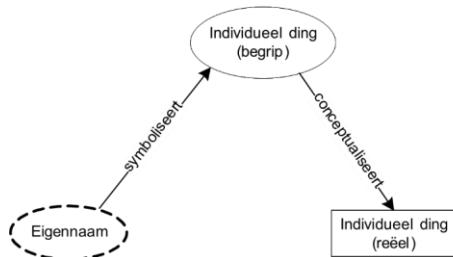
De functie van symbolen is dat ze in de communicatie kunnen worden gebruikt om naar een begrip te verwijzen. Voorwaarde is echter dat de communicerende partijen het symbool kennen. Als de ontvanger het gebaar voor Catshuis niet kent, dan heeft het voor de zender ook geen zin om dit gebaar te gebruiken.

Een symbool kan in isolatie bestaan. Een aantal vrienden kan afspreken om vier opgestoken vingers als geheim symbool voor het Catshuis te gebruiken. Veelal maken symbolen en tekens echter onderdeel uit van een tekensysteem. Elke taal is zo’n tekensysteem. De relatie tussen een bepaald tekensysteem en een bepaald conceptueel systeem heet taalbinding.

In het voorbeeld van het Catshuis wordt de eigenaam van dit gebouw gebruikt als identificerend teken. In figuur 4 wordt dit geïllustreerd.

<sup>3</sup> Merk op dat elk symbool een materiële verschijningsvorm moet hebben om te kunnen functioneren. Dit maakt dat elk symbool uiteindelijk ook een ding is. Een kaart bijvoorbeeld, die een stuk van de wereld weergeeft en onze gedachten daarover symboliseert, is in welke vorm dan ook (papier, CD-ROM, USB-stick) een ding.

<sup>4</sup> Merk ook op dat figuur 3 een symbool is van de semiotische driehoek, of preciezer: een gedachte aan de semiotische driehoek. Dus ook in het denken en schrijven over de semiotische driehoek passen mensen die toe. Zelfreferentie heet dat. Een voorbeeld van een aan zichzelf refererende zin is ‘Deze zin geen werkwoord.’

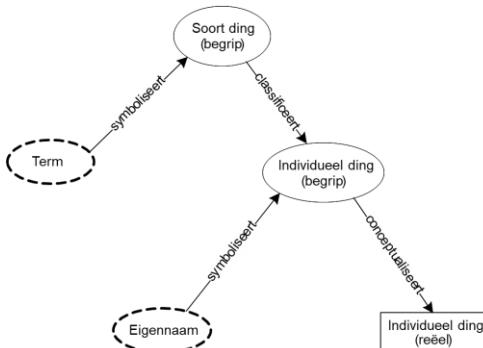
**Figuur 4 — Relaties tussen eigennaam, begrip en individueel ding**

Het Catshuis, het Mauritshuis, Mon Repos, Eben Haézer zijn allemaal namen van individuele gebouwen of huizen. Maar daarnaast kennen we ook het algemene begrip 'gebouw'. Dit correspondeert echter niet met één bepaald aanwijsbaar fysiek object. Men kan voor het Catshuis gaan staan en zeggen: "Kijk, dit is het Catshuis." Maar er bestaat geen enkel ding waarvan men kan zeggen: "Kijk, dit is het gebouw." Men kan alleen maar een reëel exemplaar aanwijzen en daarvan zeggen: "Kijk, dit is een gebouw", wat eigenlijk wil zeggen: "Kijk, dit is een voorbeeld van een gebouw."

Het betreft hier dus een begrip dat geen directe relatie heeft met een ding uit de echte wereld. Het heeft echter wel een relatie met de gedachten die van deze individuele objecten kunnen worden gemaakt: de concepten. We kunnen immers zeggen dat het Catshuis een gebouw is. Daarmee zeggen we dat het Catshuis behoort tot een verzameling die wordt aanduid met 'gebouw'. Zeggen dat het Catshuis een gebouw is, wordt classificeren genoemd. Het Catshuis wordt dan als het ware toegekend aan een bepaalde klasse (verzameling).

Naast 'gebouw' zijn er ook begrippen als 'hond', 'boom', 'brug'. Elke taal kent tienduizenden concepten. En dat zijn alleen nog maar begrippen die door één enkele term worden weergegeven. Als samengestelde termen worden meegerekend (jachthond), dan zijn het er nog veel meer.

Een eigennaam verwijst naar een (concept van) een individueel object. Op dezelfde manier verwijst een bepaalde term naar een soort object. In figuur 5 is dit weergeven door een 2<sup>e</sup> driehoek bovenop de driehoek van figuur 4 te tekenen. De classificatierelatie tussen soort ding en het individueel ding vervangt de conceptualisatierelatie tussen het conceptuele individueel ding en het reële individueel ding. Deze NTA hanteert het onderscheid tussen individueel ding en soort ding.

**Figuur 5 — Classificatierelatie tussen individueel ding en soort ding**

### *Immateriële objecten*

De voorbeelden van het Catshuis en het begrip ‘gebouw’ suggereren dat het bij ‘dingen’ altijd om fysieke en materiële objecten zou gaan. Dat is echter niet zo.

De werkelijkheid bestaat immers uit meer zaken dan alleen fysieke objecten. Ook activiteiten en immateriële zaken horen daartoe.

Pinkpop bijvoorbeeld is geen fysiek object maar een evenement. Hetzelfde geldt voor Koningsdag. Ook dit is een evenement. Op kleinere schaal geldt dat voor ‘het bakken van een pizza’ en het ‘kappen van een boom’. Dit zullen we niet zo snel evenementen noemen en eerder activiteiten. Maar dat is slechts een schaalverschil, geen wezenlijk verschil. Het is als het verschil tussen stad en dorp.

Ook hier speelt het onderscheid tussen individuen en typen. Koningsdag 2019 is immers niet hetzelfde als Koningsdag 2018 en Koningsdag 2017. Dit zijn individuele evenementen die allemaal instanties van het type Koningsdag zijn.

Ook hier kunnen subklassen en superklassen worden gevormd. Koningsdag, Koninginnedag, Bevrijdingsdag, 1<sup>e</sup> Paasdag, enz., zijn allemaal subtypen van Feestdag.

‘Dingen in de werkelijkheid’ kunnen ook immaterieel zijn.

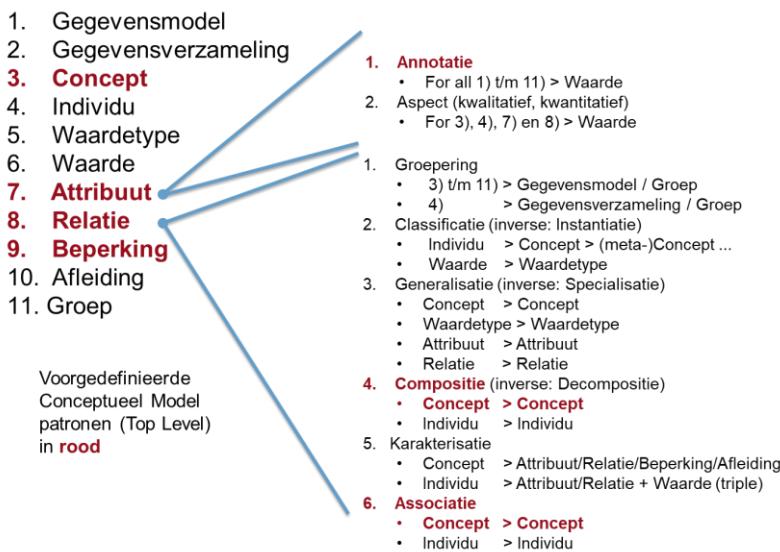
**VOORBEELD** Een kadastral perceel is weliswaar een ruimtelijk gebied, maar het bezit geen materiële kenmerken. Het is een gebied waarop één of meer zakelijke rechten rusten. Een zakelijk recht (vruchtgebruik, hypothek, enz.) is echter immaterieel.

Een topografisch perceel als grasland is daarentegen materieel. Het kenmerk waardoor het een ruimtelijke eenheid is (gras), is immers een materieel iets.

## **6.2 Conceptueel Meta Model**

### **6.2.1 Algemeen**

Mede op basis van NEN-ISO 16354 zijn in deze NTA metaconcepten gedefinieerd die samen het op RDF-gebaseerde conceptuele metamodel vormen, dat onafhankelijk is van de vier talen SKOS, RDFS, OWL en SHACL, die in hoofdstuk 7 worden besproken. Kenmerkend voor RDF is het gebruik van binaire, gerichte relaties. Figuur 6 geeft een overzicht van het CMM:



Figuur 6 — Conceptueel Meta Model (CMM)

### 6.2.2 Gegevensmodel (Data model)

Bij aankomst in een hotel moet een gast meestal een aantal gegevens opgeven: naam, huisadres, woonplaats, paspoortnummer, enz. Het hotel gebruikt daarvoor een formulier waarop deze gegevens (met pen) moeten worden ingevuld.

De voorgedrukte tekst op dit formulier kan worden beschouwd als de representatie van een gegevensmodel. Het bepaalt namelijk de gegevens die kunnen worden ingevuld. Als bijvoorbeeld het invulveld 'geboortedatum' ontbreekt, dan kan deze ook niet worden ingevuld. Dit kenmerk maakt dan geen onderdeel uit van het gegevensmodel dat het hotel gebruikt om zijn gasten te registreren.

Een gegevensmodel kan dus worden beschouwd als een afgebakende structuur ten behoeve van een geordende gegevensverzameling die nodig is voor een bepaalde toepassing. Voor de gasterregistratie van een hotel is dat duidelijk een andere verzameling dan voor de inspectie van een brug.

Vandaar dat er in de praktijk ontelbaar veel verschillende gegevensmodellen worden gebruikt, allemaal met hun eigen doel. Gegevensmodellen kunnen zwak of sterk zijn, al naar gelang de modelleerkracht. Een sterk gegevensmodel is een ontologie, of OTL, die ook beperkingen op de data vastlegt.

### 6.2.3 Gegevensverzameling (Data set)

Naast de gegevensmodellen zijn er gegevensverzamelingen (Engels: data sets), die individuele data bevatten volgens het gegevensmodel.

#### 6.2.4 Concept (Concept)

Een concept is min of meer synoniem met de term 'begrip'. Het betreft de mentale voorstelling van 'iets uit de werkelijkheid'.

Bij de behandeling van de betekenis driehoek (zie figuur 3) wordt de term 'concept' zowel gebruikt voor gedachten over individuele dingen (Catshuis) als over soorten dingen (gebouw).

Binnen het CMM is de term 'Concept' (met een hoofdletter) echter gereserveerd voor de mentale representatie van soorten dingen. Individuele dingen heten daar Individu.

**VOORBEELD** Onder Concept vallen dus zaken als brug, hond, eik, processierups, schip, windmolen, maar ook zaken als feestdag, verkeersongeval, geologisch tijdperk, schaakstelling, testament.

Belangrijk kenmerk van een Concept is dat het kan worden geïnstantieerd. Dat betekent dat er een classificatierelatie moet kunnen bestaan tussen een Individu en een Concept.

Om die reden is een begrip als 'gebouw' een Concept. Er zijn immers individuele dingen als Catshuis, Ons Nest, Eben Haëzer, enz. die kunnen worden geclasseerd als 'gebouw'. Dit betekent dat 'gebouw' kan worden gezien als een verzameling gelijksoortige Individuen.

**OPMERKING** Er kan onderscheid worden gemaakt in 'rigid'-concepten (ook wel 'typen') en 'non-rigid'-concepten (ook wel 'rollen'). Als iets is geclasseerd naar een rigid-concept, dan is dat iets dat altijd (voorbeeld 'persoon' of 'fysiek object'). Als iets is geclasseerd naar een non-rigid-concept, dan kan dat tijdelijk zijn: vanaf een bepaalde tijd (Vader) of voor een bepaalde duur (Passagier, Student).

In het bijzonder bij specialisatie is extra aandacht nodig bij het relateren van rigid- en non-rigid-concepten. Specialisatie van een rigid-concept in een non-rigid-concept gaat goed zolang er 'behoud van identiteit is'. Dat wil zeggen dat elke instantie van het non-rigid-concept overeenkomt met precies één instantie van het rigid-concept. 1) en 2) beschrijven twee situaties waarbij deze identiteit kan worden verstoord:

- 1) Er zijn meerdere onderling disjuncte super-rigid-concepten voor het non-rigid-concept in dezelfde ontologie. Voorbeeld: Vader is een rol van Persoon maar ook van Dier (Persoon en Dier onderling disjunct, iets is niet een persoon en een dier tegelijk). Een instantie van een Vader is de een of de ander, dus in elk geval heeft één superconcept geen tegenhanger. De identiteitslink is dan verstoord. Op dat moment kan een patroon worden toegepast waarbij de rollen worden gesplitst in een PersoonVader-rol en een DierVader-rol die beide kunnen worden gegeneraliseerd in een generieke Vaderrol. Dan gaat het weer goed.
- 2) Er is vanuit de definitie van het non-rigid-concept sprake van een-op-n-relatie met het rigid-concept (in plaats van een benodigde een-op-eenrelatie voor identiteitsbehoud). Goed voorbeeld is Passagier. Een Persoon speelt typisch meer dan een keer de rol van passagier (wanneer gebonden aan een vlucht), dus niet elke instantie van passagier is precies één instantie van persoon. Voor Vader gaat dit wel goed, omdat we typisch niet definiëren dat een persoon meerdere vaderrollen vervult (bijvoorbeeld per kind), maar gewoon een vaderrol voor een of meer kinderen. Student kan wel problematisch zijn als gedefinieerd 'per opleiding'. Een persoon vervult dan meerdere studentrollen. Als Student gedefinieerd is als iemand die een of meer opleidingen volgt, dan gaat het weer wel goed.

#### 6.2.5 Individu (Individual)

Onder een Individu verstaan we een begrip dat overeenkomt met 'iets uit de werkelijkheid'.

## NTA 8035:2020

**VOORBEELD** Voorbeelden zijn het al eerder genoemde Catshuis, maar ook zaken als Koningsdag 2019, de uitbarsting van de Tambora en 'Testament van een student' van P. Paaltjens.

Vaak kan naar Individuen worden verwezen met een eigenaam. Maar noodzakelijk is dit niet. Niet elk koffiekopje of elke tafel is voorzien van een eigenaam.

Maar, en dat is belangrijk, een Individu kan wel worden aangeduid. Er kan immers worden gesproken over het "3<sup>e</sup> koffiekopje rechts op de salontafel in de eetkamer van het Catshuis". Soms zijn er veel woorden voor nodig, maar in principe kan elk Individu op deze manier worden aangeduid.<sup>5</sup> Merk op dat een Individu niet meer kan worden geïnstancieerd.

### 6.2.6 Waardetype (Value type)

Het begrip Waardetype moet in combinatie worden gezien met het begrip Waarde. Een Waarde verhoudt zich tot Waardetype zoals een Individu zich verhoudt tot Concept. Een Waarde is een instantie van een bepaald Waardetype en wordt erdoor geklassificeerd.

*Wat is het verschil tussen een Waardetype en een Concept?*

Vergelijk de volgende twee zinnen:

- Amsterdam heeft 900.000 inwoners.
- Amsterdam heeft negen letters.

De naam 'Amsterdam' speelt in elk van deze twee zinnen duidelijk een andere rol. De eerste zin doet een uitspraak over de aan het IJ gelegen hoofdstad van het land. De term 'Amsterdam' wordt hier gebruikt om naar die stad te verwijzen. Deze term wordt daarom de naam van de stad genoemd. Andere verwijzers zijn 'A'dam' en 'Mokum'. De tweede zin doet een uitspraak over de term 'Amsterdam' zelf en benoemt het aantal letters. Engelsen duiden dit verschil aan met 'use' en 'mention': 'gebruiken' en 'benoemen'.

Een belangrijk verschil tussen een Concept en een Waardetype is dat een (instantie van een) Waardetype geen Toestand kan hebben (of beter: dat de Toestand niet kan veranderen in de tijd), en bij een Concept wel. Een Waardetype is dus een 'Toestandsloos' Concept.

#### Basisindeling

De basisindeling die bij waardetypen meestal wordt gehanteerd en die het CMM overneemt, is de volgende:

- tekenreeks ('character string');
- getal, onder te verdelen in:
  - natuurlijk getal (positieve integer);
  - geheel getal (integer);

<sup>5</sup> Dit houdt in dat Concepten (typen) niet direct kunnen worden aangeduid. Men kan niet zeggen: "Kijk dit is het begrip 'hond'." Men kan naar een voorbeeld wijzen ("Kijk, een hond"), maar dan wordt een Individu gebruikt om naar het type te verwijzen. Ook kan men een plaatje van een hond aanwijzen, maar ook dat is een indirecte verwijzing (er bestaan namelijk veel plaatjes van honden).

- rationaal getal in tientallige notatie (decimale waarde);
- reëel getal.
- waarheidswaarde (boolean-waarde);
- diverse datum- en tijdgerelateerde datatypen.

#### *Tekenreeks*

Een tekenreeks is een rij achter elkaar geplaatste lettertekens. Voorbeelden zijn: atg, xxrgt, hond, enz. Vaak spreekt men af om alleen de letters uit het Latijnse alfabet te gebruiken. Maar er kan ook worden afgesproken om tevens cijfers toe te laten zodat reeksen mogelijk zijn als ftr8s2a. Ook kunnen nog andere (grafische) tekens worden toegelaten zodat er reeksen mogelijk zijn als gh&54%d. Uitbreiding met tekens uit andere alfabetsystemen is ook nog mogelijk. Het is maar wat er met elkaar wordt afgesproken.

#### *Getal*

Een getal is in principe een reëel getal, maar er kan ook worden afgesproken om alleen gehele of natuurlijk getallen te gebruiken. De verzameling van natuurlijke getallen is immers een subset van die van de reële getallen.

Let wel, een tekenreeks als 12 kan niet automatisch worden beschouwd als een getal. Het is in eerste instantie een 1 gevolgd door een 2. Pas als er wordt afsGEproken om deze reeks te beschouwen als een notatie volgens het decimale stelsel, kan 12 worden geïnterpreteerd als het getal twaalf.

#### *Waarheidswaarde*

Waarheidswaarde is een waardetype met slechts twee mogelijke waarden, 'true' (waar, ja) en 'false' (onwaar, nee). Het wordt veel gebruikt in computerprogramma's, maar ook ingezet bij dataregistraties. Als er een 'vinkje' in een formulier wordt gezet, dan wordt in feite dit waardetype gebruikt.

#### *Onderverdelingen*

De verzameling van alle mogelijke tekenreeksen is oneindig groot. Hetzelfde geldt voor de verzameling van alle getallen. Vaak bestaat echter de wens om voor een specifieke toepassing een kleine specifieke deelverzameling te gebruiken.

VOORBEELD 1 Nederland heeft twaalf provincies. In veel registraties wordt naar een bepaalde provincie verwezen met een tweelettercode. Van de 676 mogelijke tweelettercodes zijn er echter slechts twaalf nodig: 'fr' en 'gr' zijn geschikte codes, 'xl' en 'st' echter niet.

Om deze specifieke deelverzameling te kunnen onderscheiden van andere deelverzamelingen wordt deze verbonden met een specifieke naam: Provinciecode bijvoorbeeld, of nog specieker: tweeletterige Provinciecode (want er bestaat ook, geloof het of niet, een tweecijferige Provinciecode).

VOORBEELD 2 Getallen zijn heel handig als codes voor begrippen die een inherente ordening hebben. NEN 2767-reeks bijvoorbeeld hanteert de conditiescores: uitstekend, goed, redelijk, matig, slecht en zeer slecht. Deze begrippen hebben een zekere ingebakken orde. Daarom kunnen ze op natuurlijke wijze worden verbonden met de getallen 1 t/m 6, omdat de natuurlijke getallen ook dezelfde ingebakken orde hebben. Om dezelfde reden wordt deze specifieke deelverzameling verbonden aan een specifieke naam: Conditiescorecode of iets dergelijks.

#### *Waardetype en Waarde*

## NTA 8035:2020

In de voorbeelden 1 en 2 kunnen de tweeletterige Provinciecode en de Conditiescorecode worden beschouwd als Waardetype. Een individuele code als 'fr' en '2' is dan een Waarde binnen dat specifieke type.

### 6.2.7 Waarde (Value)

Waarden zijn instanties van een bepaald Waardetype. Waarden kunnen tekst-strings zijn maar ook bijvoorbeeld numeriek, zoals integers, decimalen of waarheidswaarden (waar, onwaar).

Bij numerieke waarden is het belangrijk om de gehanteerde eenheid te noemen. Als men zegt: "Het was vandaag 23 graden", dan kan men de vraag stellen: Fahrenheit, Celsius, Réaumur of Kelvin? Want dat zijn de eenheden waarin de temperatuur kan worden gemeten.

Deze kwestie speelt bij waardetypen waarbij de individuele waarden worden voorgesteld door een getal. Hierbij volgt een nadere toelichting:

- Bij het waardetype Huisnummer bijvoorbeeld speelt het geen rol. Bij huisnummer 21 heeft het getal 21 een puur identificerende functie. Het is immers niet zo dat het huis met huisnummer 22 groter is dan dat met huisnummer 21. Ze zijn alleen verschillend.
- Bij grootheden speelt dit wel. 23 leerlingen in een klas is meer dan 20 leerlingen. Een tafel van 2,70 m is langer dan een tafel van 2,30. 28 °C is warmer dan 18 °C.
- Vaak is echter de teleenheid vanzelfsprekend en hoeft deze niet nader te worden toegelicht. Als het aantal leerlingen in een klas moet worden bepaald (de klasse-grootte), dan is het eigenlijk niet nodig om aan te geven dat de enkele leerling de teleenheid is. Er is niemand die het in zijn hoofd haalt om bijvoorbeeld 1,78 of  $\pi$  leerlingen als teleenheid te kiezen.
- Bij grootheden als Lengte en Temperatuur is het waardebereik echter continu. Dat wil zeggen dat het elk reëel getal als waarde kan hebben. Het waardebereik is niet beperkt tot bepaalde discrete waarden zoals aantal leerlingen.
- Bij deze grootheden is de teleenheid niet vanzelfsprekend en moet daarom worden afgesproken. En omdat er in het verleden voor deze grootheden verschillende afspraken zijn gemaakt (meter, 'foot'), is het vaak nodig om aan te geven welke eenheid er wordt gebruikt. Ook bij gebruik van dezelfde basiseenheid kunnen er nog verschillende varianten worden gebruikt: mm, cm, m, km, enz.

### 6.2.8 Attribuut (Attribute)

Vervolgens zijn er Attributen die intrinsieke kenmerken beschrijven.

Attributen kunnen worden onderverdeeld in annotaties, door mensen te interpreteren 'identifiers', namen, labels, enz., en door computers te interpreteren kwalitatieve en kwantitatieve aspecten.

Een attribuut lijkt dus op een relatie maar verbint nu een individu met een waarde in plaats van een referentie naar een ander individu.

#### *Annotatie (Annotation)*

Annotaties kunnen worden gebruikt om een willekeurige 'vrije tekst' te koppelen aan een bepaalde instantie van een metaconcept.

Stel dat men wil zeggen dat van het concept Tuibrug meer informatie is te vinden in een bepaalde publicatie, op een bepaalde internetpagina, of dat men wil zeggen dat de definitie afkomstig is van

T. Gerritsen. Al deze informatie kan in de vorm van een Annotatie aan het concept Tuibrug worden gerelateerd. Annotaties zijn bestemd voor menselijke interpretatie.

*Aspect (Aspect)*

Een aspect daarentegen is een door de computer te interpreteren attribuut, kwalitatief of kwantitatief. Kwantitatieve aspecten zijn gerelateerd aan grootheden en vragen om een eenheid bij de waarde.

**6.2.9 Relatie (Relation)**

In de wiskunde beschrijft een relatie het verband of de betrekking tussen objecten. Elke relatie is gedefinieerd over een aantal verzamelingen en verbindt, uit deze verzamelingen, de elementen die met elkaar in het bedoelde verband staan. Het aantal verzamelingen waarover de relatie is gedefinieerd, heet de plaatsigheid of ariteit van de relatie. De relatie is één van de centrale begrippen uit de wiskunde. De meest voorkomende relatie is de tweeplaatsige relatie, die objecten in tweetallen aan elkaar koppelt. (Bron: *Wikipedia*.)

Relaties die tweeplaatsig zijn, worden ook binaire relaties genoemd.<sup>6</sup> <sup>7</sup>

Het CMM onderscheidt zes subrelaties, die hier kort worden toegelicht:

- groepering;
- classificatie;
- generalisatie;
- compositie;
- karakterisatie;
- associatie.

---

<sup>6</sup> Relaties met een hogere ariteit kunnen in principe worden ontbonden in een serie van gestapelde relaties (objectificatie).

<sup>7</sup> In principe bestaan er ook eenplaatsige (unaire) relaties. In het CMM komt deze relatie overeen met Concept.

*Groepering (Grouping)*

Deze relatie kan worden gebruikt om aan te kunnen geven welke Concepten, Waardetypen en andere gegevenstypen bij een Gegevensmodel horen.

VOORBEELD 1 Zo kan bijvoorbeeld worden gezegd dat het Gegevensmodel Hotelreceptie bestaat uit: voornaam, achternaam, straatnaam en huisnummer, postcode en woonplaats, land, paspoortnummer en (indien van toepassing) kenteken voertuig.

*Classificatie (Classification)*

*Inverse: Instantiatie (Instantiation)*

Classificatie wordt gebruikt om de betrekking tussen een Individu en een Concept aan te geven.

VOORBEELD 2 Deze relatie wordt bijvoorbeeld gebruikt als er wordt gezegd dat het Catshuis een Gebouw is. Het Catshuis is hier een Individu en Gebouw is een Concept.

De classificatierelatie wordt ook gebruikt om de links naar het metamodel te maken.

VOORBEELD 3 rws:Brug rdf:type owl:Class.

De classificatierelatie wordt ook gebruikt om de relatie tussen een Waarde en een Waardetype aan te geven.

VOORBEELD 4 Bijvoorbeeld als er wordt gezegd dat Hout een Materiaalwaarde is of dat 19 graden Celsius een Temperatuurwaarde is.

*Generalisatie (Generalisation)*

*Inverse: Specialisatie (Specialisation)*

Generalisatie is het terrein van subtypen (of subklassen) en supertypen (of superklassen).

VOORBEELD 5 Zo kan bijvoorbeeld worden gezegd dat elke hond een zoogdier is. Dat impliceert dat de verzameling van alle honden een deelverzameling is van de verzameling van alle zoogdieren. 'Zoogdier' wordt dan een generalisatie of supertype van 'hond' genoemd. Andersom wordt 'hond' een specialisatie of subtype van 'zoogdier' genoemd.

Generalisatierelaties kunnen niet alleen worden gedefinieerd tussen Concepten (zoals tussen 'hond' en 'zoogdier'), maar ook tussen Attributen en Relaties. De indeling van Relatie in de subtypen die in deze paragraaf worden behandeld, is daarvan een voorbeeld.

Met name de associaties, maar ook de karakterisaties kunnen nog verder worden onderverdeeld.

VOORBEELD 6 Zo zijn de werkwoorden zagen, schaven en schuren subtypen van het werkwoord bewerken. Dit houdt in dat de populatie van de associatie 'Werknemer X schaft Raamkozijn Y' een subset is van de populatie van de associatie 'Werknemer X bewerkt Raamkozijn Y'.

Merk op dat de subtyperelatie tussen Concepten en tussen associaties vaak afhankelijk van elkaar zijn en uit elkaar kunnen worden afgeleid.

VOORBEELD 7 Zo kan de associatie 'Persoon X is de eigenaar van zoogdier Y' worden afgeleid uit de associatie 'Persoon X is de eigenaar van Hond Y' als bekend is dat het concept Hond een subtype is van het concept Zoogdier.

*Compositie (Composition)*

*Inverse: Decompositie (Decomposition)*

Compositie is het terrein van de deel-geheelrelaties.

VOORBEELD 8 Zo kan bijvoorbeeld worden gezegd dat de tafel in de keuken van het Catshuis bestaat uit een tafelblad en vier tafelpoten. De tafel is het geheel, het tafelblad en de vier poten zijn de onderdelen.

Merk op dat compositierelaties worden beschreven tussen Individuen, niet tussen Concepten. Het is een individuele tafel die bestaat uit een individueel tafelblad en vier individuele poten.

Compositie-uitspraken die worden geformuleerd tussen Concepten, betekenen echter iets anders. Als er bijvoorbeeld wordt gezegd dat een tafel bestaat uit een tafelblad en drie of vier tafelpoten, dan betekent dat niet dat het begrip 'tafel' is opgebouwd uit het begrip 'tafelblad' en vier- of driemaal het begrip 'tafelpoot'.

Het kan onder meer worden opgevat als een definitie: een ding dat bestaat uit een ding dat 'tafelblad' heet en uit drie of vier andere dingen die 'tafelpoot' heten, kan worden geklassificeerd als 'tafel'. Niets anders is een tafel.

Het kan echter ook worden gelezen als een productspecificatie: elke individuele tafel die gemaakt gaat worden, moet uit een tafelblad en uit drie of vier tafelpoten bestaan. Als dat niet zo is, dan wordt het product afgeweerd. Deze kennis wordt vastgelegd met beperkingen.

*Karakterisatie (Characterisation)*

Dit zijn relaties die gelden (relevant zijn) tussen een Individu en een attribuutwaarde of een referentie naar een ander individu. Men kan bijvoorbeeld zeggen dat de Domtoren een hoogte heeft van 110 m. Hierbij wordt het individu 'de Domtoren' gekarakteriseerd door de waarde 110 m voor het attribuut hoogte. De karakterisatie zit bij een RDF-binding impliciet in de 'triple': een subject wordt gekarakteriseerd door een verzameling predicaat-objectparen.

Afhankelijk van de gekozen taal (SKOS, RDFS, OWL, SHACL) is karakterisatie ook relevant op conceptniveau. Bij RDFS en OWL geldt door het Open World Assumption (OWA) dat elke property (attribuut of relatie in de termen van deze NTA) in principe kan worden gebruikt bij elke 'class' (concept). Kortom, een mogelijke karakterisatie op klasse-niveau is hier niet relevant.<sup>8</sup>

De beperkingen worden bij OWL geregeld via specialisatie (de klasse is een subklasse van alle dingen waarvoor de restrictie geldt). Bij SHACL (CWA) is karakterisatie wel relevant. Properties (attributen en relaties) kunnen hier (ook niet beperkt) worden gekoppeld aan SHACL shapes via sh:property.

*Associatie (Association)*

Iets soortgelijks geldt voor Associaties. Dit zijn ook relaties die worden beschreven tussen twee (verschillende) Individuen.

VOORBEELD 9 Een voorbeeld is de uitspraak: "Jan beheert het Catshuis." Hier wordt het individu 'Jan' geassocieerd met het individu 'Catshuis'. De associatie wordt hier geklassificeerd als 'beheren'.

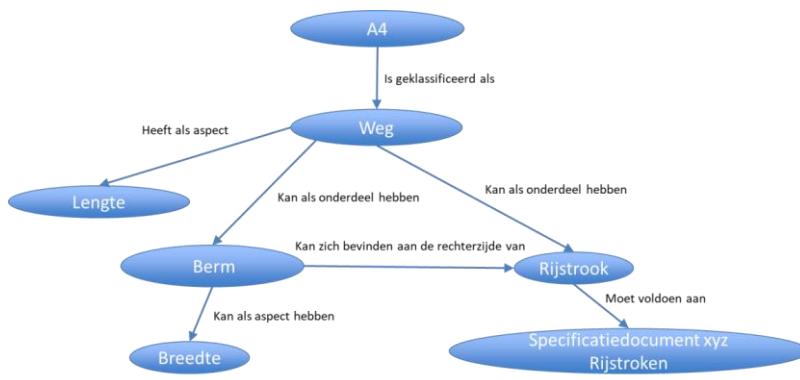
De associatie-uitspraken die tussen Concepten worden geformuleerd, hebben een andere betekenis en een andere functie. Men kan bijvoorbeeld zeggen dat een Persoon één of meer Gebouwen kan beheren. Deze uitspraak legt een beperking op de mogelijke invullingen van de individuen die hieraan kunnen

<sup>8</sup> Soms wordt nog wel eens expliciet 'min kardinaliteit = 0' gemodelleerd om de relevantie en dus karakterisatie aan te geven. Dit is echter oneigenlijk.

voldoen. Dit betekent bijvoorbeeld dat Jan een Persoon moet zijn (en bijvoorbeeld geen kanarie) en het Catshuis een Gebouw (en bijvoorbeeld geen IJsberg). Dit wordt dus geregeld via beperkingen.

#### **Netwerken van concepten, individuen, eigenschappen en relaties**

Een relatie met zijn verwante concepten (respectievelijk 'links' en 'rechts' van de relatie) vormt samen de uitdrukking van een enkel (atomair) feit, zijnde een klein stukje 'kennis'. Een dergelijke verzameling van feitenuitingen vormt een kennisnetwerk (ook wel 'graaf'), dat een specifiek kennisgebied over het soort dingen uitdrukt. Dit wordt geïllustreerd in figuur 7 (toevallig een voorbeeld waarbij de eigenschappen 'geobjectificeerd' zijn, dat wil zeggen, als klasse gemodelleerd, maar dit terzijde).



**Figuur 7 — Voorbeeld van een gegevensmodel voor een weg**

Figuur 7 illustreert het kennismodel samengesteld uit een aantal specifieke atomaire feiten, zoals het feit dat een rijstrook deel kan uitmaken van een weg.

Elk van deze feiten wordt uitgedrukt door een relatie van een bepaald type. De verschillen tussen verschillende gegevensmodellen zijn vooral te wijten aan het feit dat die relatietypen vaak niet explicet worden gemaakt en dat er weinig standaardisatie is van die relatietypen.

#### **6.2.10 Beperking (Constraint)**

Concepten kunnen Beperkingen hebben die het aantal waarden of de waarden **zelf** beperken. Ook attributen en relaties kunnen beperkingen hebben met betrekking tot hun bron-, doelconcepten (in het geval van relaties) of waardetype (in het geval van attributen).

**VOORBEELD** Een voorbeeld van een beperkingsregel is: "Een vereniging kan (op een bepaald moment) hooguit één voorzitter hebben."

Het voorbeeld is een regel die wordt geformuleerd op typenniveau. Er wordt immers over verenigingen en voorzitters in het algemeen gesproken en niet over een specifieke vereniging en een specifiek persoon.

Maar de regel geldt op individueel niveau. Het legt de relaties die een specifieke vereniging kan hebben immers aan banden. Stel dat T. Gerritsen de voorzitter is van de schutterij St. Sebastianus. Als P. Hendriksen vervolgens beweert dat hij óók de voorzitter is van St. Sebastianus, dan komen deze twee feiten in conflict met de regel die in het voorbeeld is geformuleerd.

Er bestaan in de praktijk veel verschillende soorten beperkingsregels. De meest bekende en de meest voorkomende zijn de zogeheten kardinaliteitsregels, waarvan de voorzittersregel een voorbeeld is. Standaard ('default') geldt namelijk dat binaire relaties bi-directioneel [0:m] zijn. Door te stellen dat deze relatie [1:n] is in plaats van [n:m] in een bepaalde richting wordt het aantal mogelijke instanties ingeperkt. Vandaar de naam 'beperkingsregel'.

Beperkingen kunnen worden onderverdeeld in definiërende beperkingen en specificerende beperkingen. De eerste zijn per definitie altijd waar/relevant. De tweede zijn waar/relevant met een specifiek doel voor ogen.

Het feit dat een auto een motor heeft, is een definiërende beperking, terwijl de eis dat de motor minstens 200 pk moet hebben, een specificerende beperking is. De specificerende beperkingen zijn meestal afkomstig van een klant of van regelgeving.

Het onderscheid is altijd relatief ten opzichte van een specifiek concept. Voor een Brug kan 'materiaal is staal', een specificerende beperking zijn, maar voor een StalenBrug is het een definiërende beperking. Het mechanisme voor specialisatie is nauw verwant: een subklasse is meestal een superklasse plus een of meer definiërende beperkingen die betrekking hebben op attributen en/of relaties die nog niet beperkt zijn voor de superklasse.

Er zijn nog veel meer soorten beperkingsregels, maar de behandeling daarvan valt buiten het toepassingsgebied van deze NTA. Hiervoor wordt verwezen naar tekstboeken op het gebied van informatiemodellering, zoals bijvoorbeeld [1] en [2].

#### **6.2.11 Afleiding (Derivation)**

Er zijn Afleidingen die vertellen hoe nieuwe waarden voor attributen of referenties voor relaties kunnen worden afgeleid uit bestaande waarden/referenties.

**VOORBEELD** Als Maria de moeder is van Piet en Piet de vader is van Jordan, dan kan daaruit worden afgeleid dat Maria één van de twee grootmoeders is van Jordan.

De afleiding uit het voorbeeld kan worden gemaakt omdat het hebben van familierelaties algemene kennis is, zoals verwoord in de volgende regels:

- 1) Als A de moeder is van B en B de ouder van C, dan is A de grootmoeder van C.
- 2) C heeft precies één (biologische) vader en één (biologische) moeder.
- 3) Ouder = vader of moeder.

Door deze regels te formaliseren (om te zetten in computerverwerkbare code) wordt dit onderdeel van een kennissysteem en kan ook de computer deze afleiding maken. Merk op dat regel 2) het karakter heeft van een beperkingsregel. Deze regels kunnen dus ook bij een Afleiding worden gebruikt.

Regels 1) en 3) kunnen definities worden genoemd. Maar ook een definitie is een soort beperking (wat al in de naam besloten is). Regel 1) zegt bijvoorbeeld dat niet iedereen de grootmoeder kan zijn van C. Alleen iemand die de moeder is van één van de twee ouders van C, kan de grootmoeder van C worden genoemd.

### 6.2.12 Groep (Group)

Naast de groepering van alle soorten meta-concepten in gegevensmodellen (individuals als referentie-individuals) en alle individuals in gegevensverzamelingen, is er ook vaak een behoefte om specifieke verzamelingen concepten, attributen, relaties of individuals etc. te kunnen definiëren.

Een goed voorbeeld is de groepering van attributen al naar gelang een bepaald product aspect bijvoorbeeld alle geometrische attributen.

## 7 Taalbinding naar W3C-standaarden

### 7.1 De moeder aller W3C LD/SW-talen: RDF

Er zijn veel beschikbare technologieën/talen voor het modelleren van gegevens en gegevensstructuren. Door deze te filteren volgens de basisprincipes 'slim' en 'standaard', ontstaat de volgende shortlist:

- ISO 10303 STEP-technologie (vanaf ~jaar 1984);
- OMG Object Oriented/Model Driven-technologie (vanaf ~jaar 1994);
- W3C XML-technologie (vanaf ~jaar 1998);
- W3C Linked Data (LD)/Semantic Web (SW)-technologie [LD] (vanaf ~jaar 2006).

In deze NTA is gekozen voor de W3C Linked Data/Semantic Web-benadering als primaire technologie. De meeste voordelen vloeien voort uit het feit dat deze benadering gebruikmaakt van internet/WWW als de onderliggende communicatie-infrastructuur. Kort gezegd: W3C nam hun bestaande WWW, zelf al gebaseerd op internet, en voegde er computerverwerkbare data aan toe (Linked Data) en maakte deze data vervolgens ook nog computerinterpreteerbaar (Semantic Web).

De W3C Linked Data/Semantic Web-benadering heeft de volgende voordelen:

- maximaal hergebruik internet/WWW (HTTP-, URI-, REST-protocollen) en gedistribueerde mogelijkheden, schaalbaar;
- solide gefundeerd in wiskunde en logica;
- 100 % generiek, breed toepasbaar;
- veel modelleerkracht (beperking, regels, enz.);
- eenzelfde taal voor gegevensstructuren en gegevens;
- gegevensstructuren zijn flexibel en uitbreidbaar;
- veel de facto, standaard 'semantische resources' beschikbaar;
- naast standaard formaten ook standaard directe gegevensbenadering (SPARQL);
- naast modellering, veel aandacht voor linking van data, meerdere waarheden-views apart te modelleren en voor te (inter)relateren netwerken van gegevens en gegevensstructuren;
- goede software-ondersteuning voor zowel ontwikkeling als gebruik;

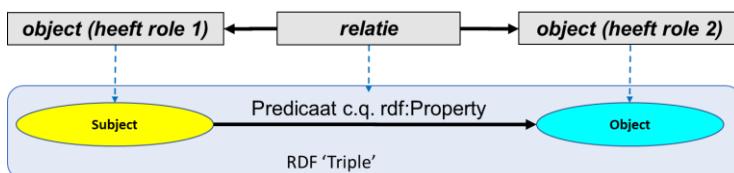
- goed beheerd en actief doorontwikkeld door W3C;
- internationaal gedragen door toonaangevende organisaties (universiteiten, kennisinstellingen, bedrijven en belangenorganisaties).

Om feiten te definiëren gebruikt deze NTA RDF. RDF is een taal voor het uitdrukken van datamodellen met behulp van statements in triple-formaten en deze te delen met andere mensen en machines. Het principe om uitspraken te gebruiken die een 'specifieke wereld' beschrijven (een ontologie), kan vaste datamodellen vervangen en biedt een uitbreidbare ontologie met referentiegegevens met het doel om kennissystemen te definiëren, aan te passen en te harmoniseren. Een feit in RDF wordt uitgedrukt door middel van een RDF-triple, die een relatie vertegenwoordigt tussen de twee dingen die het verbindt. Een triple omvat ten minste:

- een ding dat 'subject' wordt genoemd (heeft de rol van 'linkerkant');
- een ding genaamd 'object' (heeft de rol van 'rechterkant');
- een predicaat (rdf:Property) dat een (binaire) relatie aangeeft tussen een subject en een object.

De onderliggende structuur van elke expressie in RDF is dus een verzameling van drie elementen: een subject, een predicaat en een object. Een verzameling van dergelijke triples wordt een RDF-graph genoemd. Dit is te zien in figuur 8, waarbij een triple wordt beschreven als een 'node-edge-node'-koppeling. De richting-edge is relevant: deze wijst altijd naar het object met een pijl.

Zie bijlage B voor een uitgebreidere introductie van RDF.



**Figuur 8 — Gebruik van het RDF triple-principe voor feiten**

## 7.2 Toepassingstypen en benodigde modelleerkracht

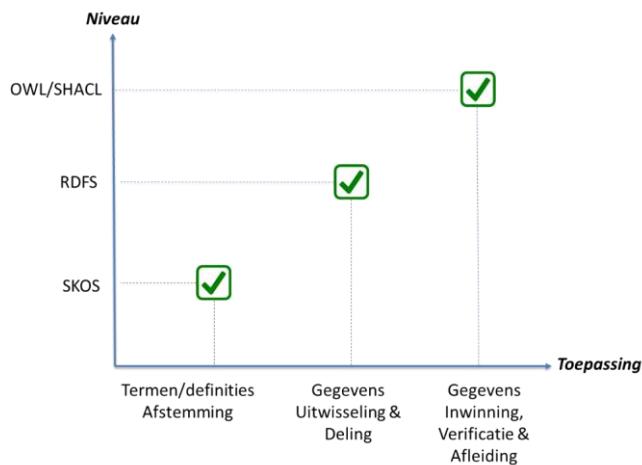
Afhankelijk van het toepassingstype (Engels: use case type) is een zwakke of sterke modelleerkracht nodig. Er zijn drie toepassingstypen:

- 1) afstemming (Engels: alignment) van termen en definities;
- 2) gegevensintegratie door uitwisseling of deling;
- 3) verificatie of afleiding van gegevens uit bestaande gegevens.

De modelleerkracht verwijst naar de mogelijkheden om automatisch door de computer gegevens te laten verwerken en interpreteren. De kracht is nauw gerelateerd aan het gehanteerde deel van het conceptuele metamodel en de bijbehorende taalbinding. Hoe completer het metamodel en hoe completer en directer de taalbinding, des te krachtiger de modellering en de resulterende gegevensmodellen en bijbehorende gegevensverzamelingen.

In de Linked Data/Semantic Web-wereld bestaan drie niveaus van modelleerkracht:

- 1) SKOS (vocabulaires, woordenboeken en thesauri);
- 2) RDFS (gestructureerde datasets en ontologieën; RDFS brengt onderscheid in concepten en individuen);
- 3) OWL/SHACL (geavanceerde ontologieën, inclusief beperkingen en regels).<sup>9</sup>



**Figuur 9 — Niveau van modelleerkracht gerelateerd aan soort toepassing**

#### **Toepassingstype 1: Afstemming van termen en definities**

De meeste eerste initiatieven op het gebied van interoperabiliteit beginnen met het afstemmen van terminologie binnen en buiten een organisatie. Organisaties die met harmonisatie beginnen, hebben te maken met veel bestaande termen en definities die meestal niet voorkomen uit het denken in een ontologie of taxonomie. Daarom is een zwakke maar flexibele taal nodig om al deze termen te identificeren, onderling te relateren en eventueel in een logische hoger/lager-hiërarchie onder te brengen. Termen en definities worden hiermee computerverwerkbaar, maar de interpreteerbaarheid blijft zeer beperkt. Dit is niet erg, want het gaat hier vaak om menselijke afstemming.

#### **Toepassingstype 2: Gegevensintegratie door uitwisseling of deling**

Tijdens de levenscyclus van een asset, van PvE, via ontwerp, uitvoering, beheer en onderhoud tot sloop, creëren veel bedrijven in verschillende rollen gegevens over die asset of delen daarvan en gebruiken deze voor verschillende doeleinden.

Bedrijven gebruiken daarvoor veel verschillende softwarepakketten en elk pakket slaat die gegevens op volgens zijn eigen formaat en gegevensstructuur. Hierdoor is gegevensuitwisseling of -deling door middel van een datalevering (hand-over) een moeilijke en kostbare onderneming geworden. Dit kan worden vergeleken met de problemen in de communicatie tussen mensen die verschillende talen

<sup>9</sup> Het verschil tussen OWL en SHACL zit hem voornamelijk in de 'open'- respectievelijk 'closed world'- interpretatie, maar SHACL is ook krachtiger dan OWL (zowel wat betreft beperkingen als afleidingen).

spreken. In dat geval is een oplossing dat er een gemeenschappelijke taal wordt gebruikt tussen betrokken ondernemingen, zoals Engels, Spaans of Frans. Een vergelijkbare oplossing kan worden gevonden voor informatie-uitwisseling tussen softwaresystemen. Die oplossing is om een gemeenschappelijke weergave van gegevens te hanteren: een afgesproken formaat (syntax) en een afgesproken structuur (semantiek).

**VOORBEELD 1** Beschouw de use case dat een klant, zoals een asset-eigenaar, zijn leveranciers in het kader van asset-management data wil laten aanleveren, voor zijn eigen informatiesystemen vereist: gegevens over ontwerpen, gebouwde en onderhouden van assets, hun samenhangende structuur (delen) en hun eigenschappen en relaties met hun omgeving, inclusief hieraan gekoppelde documenten en tekeningen.

#### **Toepassingstype 3: Verificatie en afleiding van gegevens**

Kennis over concepten, attributen en relaties wordt enerzijds vastgelegd in ontologieën door middel van toegelaten semantiek met betrekking tot een klasse en anderzijds door niet-toegelaten semantiek door middel van beperkingen en regels met betrekking tot een klasse. Om er zeker van te zijn dat uitgewisselde of gedeelde gegevens aan deze eisen en regels voldoen, zal een door een computer verwerkbaar verificatiemethode helpen om de kwaliteit van de uitgewisselde gegevens te waarborgen. Dit vereist dat beperkingen en regels zodanig worden geformaliseerd dat ze computerinterpreteerbaar zijn. Dit is waar OWL en SHACL hun toepassing vinden, bijvoorbeeld door middel van hun restricties respectievelijk 'shapes' en 'rules'.

**VOORBEELD 2** Beschouw de toepassing dat Rijkswaterstaat in een ontologie explicet een beperking wil definiëren van toegelaten waarden voor materiaal (bijvoorbeeld staal of beton) of eigenschappen van gedefinieerde objecten (bijvoorbeeld een specifieke dikte moet tussen 2 en 3 cm zijn).

**OPMERKING** In alle gevallen kan een klant eisen dat zowel de contractant als de klant zelf de oplossingsgegevens verifieert tegen de expliciete beperkingen en regels die in de ontologie zijn gedefinieerd.

In 7.3 t/m 7.6 worden de drie modelleerniveau's van figuur 9 verder gedetailleerd en gedefinieerd in termen van W3C-standaarden met hun toepasbare taalconstructies van respectievelijk SKOS, RDFS en OWL/SHACL (RDF speelt als basis overal een rol).

### **7.3 Taalbinding gebaseerd op SKOS**

SKOS staat voor Simple Knowledge Organization System. De naam SKOS werd gekozen om het doel te benadrukken van het verschaffen van een eenvoudig maar effectief gegevensmodel voor het uitdrukken van kennisorganisatiesystemen (KOS), zoals vocabulaires, woordenboeken en thesauri.

Met SKOS kan een systeem voor kennisorganisatie worden uitgedrukt in computerleesbare gegevens. Deze gegevens kunnen vervolgens worden uitgewisseld tussen computertoepassingen en gepubliceerd in een door een computer leesbaar formaat op het web.

Het SKOS-gegevensmodel maakt het mogelijk om de basisstructuur en inhoud van conceptschema's uit te drukken. Een conceptschema wordt hier gedefinieerd als: een verzameling (SKOS-) concepten, optioneel inclusief uitspraken over semantische relaties tussen deze concepten.

**OPMERKING 1** Thesauri, classificatieschema's, (zwakke) taxonomieën en andere soorten gecontroleerde vocabulaires zijn allemaal voorbeelden van conceptenschema's.

De SKOS Core-vocabulaire kan worden gebruikt in combinatie met RDF om de inhoud en structuur van een conceptschema uit te drukken als een RDF-graaf waarin concepten ondubbelzinnig worden geïdentificeerd door URI's. SKOS-gegevens worden uitgedrukt als RDF-triples en kunnen worden gecodeerd met behulp van een RDF-formaat (zoals Turtle).

## NTA 8035:2020

SKOS-concepten kunnen worden gedocumenteerd met verschillende soorten notities: scope notes, definities, redactionele opmerkingen, enz.

SKOS-concepten van verschillende conceptschema's kunnen in kaart worden gebracht. SKOS biedt vier basistypen van de koppelingshiërarchie: hiërarchisch, associatief, sterk gelijkwaardig en exact equivalent.

SKOS-concepten kunnen worden gelabeld met een willekeurig aantal lexicale strings in een bepaalde natuurlijke taal. Een van deze labels in een bepaalde taal kan worden aangegeven als het voorkeurslabel voor die taal en de andere als alternatieve of verborgen labels.

Met SKOS kunnen verschillende informatiesystemen onderling worden verbonden (geharmoniseerd) door het expliciet definiëren van gelijkwaardige elementen, aanwezig in deze kennissystemen.

De exacte XSD/RDF/SKOS-deelverzamelingen voor deze NTA zijn opgenomen in bijlage A.

De exacte mapping van het CMM naar SKOS is gegeven in figuur 10.

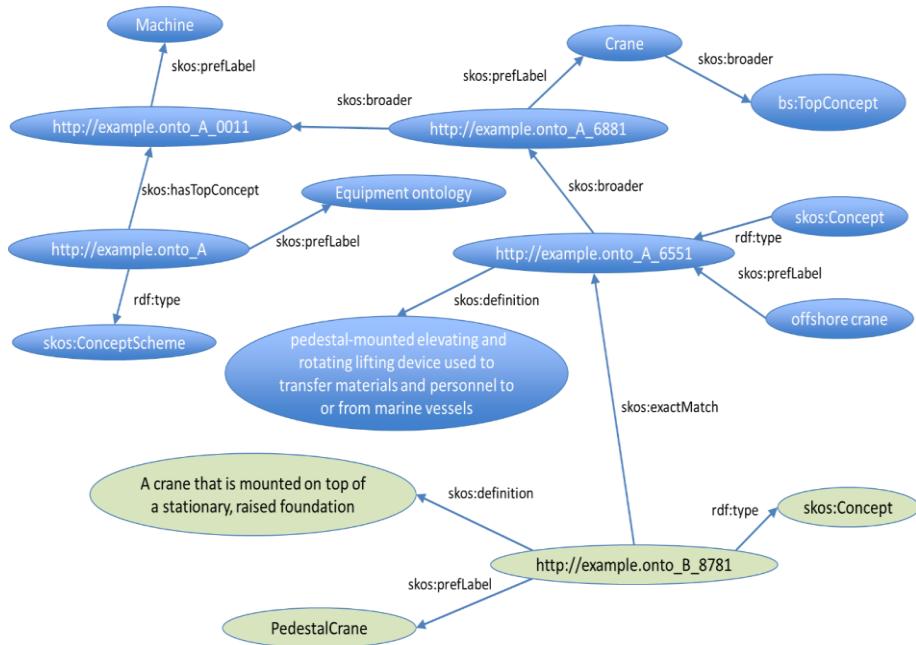
1) Gegevensmodel	>>>	skos:ConceptScheme
2) Gegevensverzameling	>>>	skos:ConceptScheme
3) Concept	>>>	skos:Concept
• top level-concepten	>>>	instanties van (rdf:type) skos:Concept met skos:isTopConceptOf relatie
4) Individu	>>>	skos:Concept
5) Waardetype	>>>	niet van toepassing
6) Waarde	>>>	alleen 'RDF plain literals' (impliciet)
7) Attribuut		
• Annotatie	>>>	skos:Concept
• top level-annotaties	>>>	instanties van (rdf:type) skos:Concept
• Aspect (kwalitatief, kwantitatief)	>>>	skos:Concept
8) Relatie		(binary & directed)
• Groepering	>>>	skos:inScheme, skos:member
• Classificatie / Instantiatie	>>>	skos:broader / skos:narrower
• Generalisatie / Specialisatie	>>>	skos:broader / skos:narrower
• Compositie / Decompositie	>>>	skos:broader / skos:narrower
• Karakterisatie	>>>	skos:related op type-niveau
• Associatie	>>>	skos:related
• top level-associaties	>>>	instanties van (rdf:type) skos:Concept
9) Beperking	>>>	niet van toepassing
10) Afleiding	>>>	niet van toepassing
11) Groep	>>>	skos:Collection

**Figuur 10 — Taalbinding naar SKOS**

Merk op dat soms in plaats van of naast skos:ConceptSchema owl:Ontology lenen van OWL. Vaak wordt SKOS bewerkt met RDFS/OWL-software tools die dit ondersteunen.

Een stappenplan voor de ontwikkeling van een SKOS-schema is opgenomen in bijlage E.

OPMERKING 2    Figuur 11 toont een voorbeeld van verbonden SKOS-schema's.



Figuur 11 — Voorbeeld van verbonden SKOS-schema's

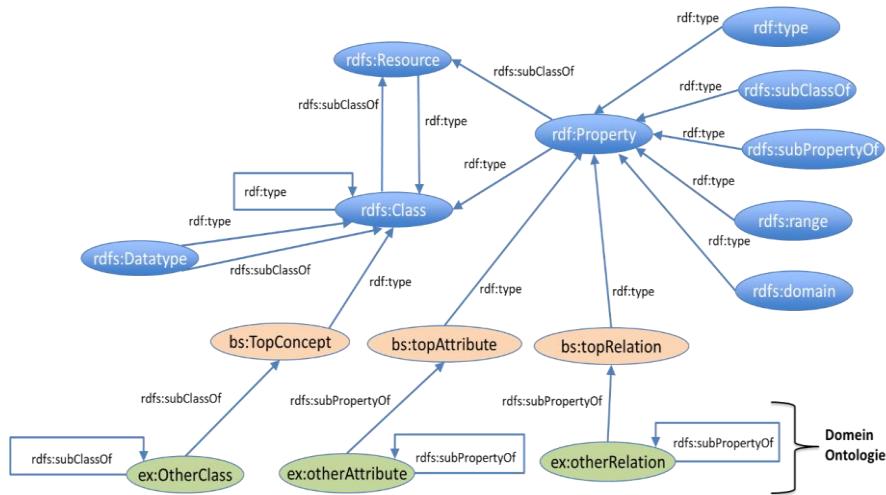
#### 7.4 Taalbinding gebaseerd op RDFS

RDFS is een uitbreiding op RDF waarbij expliciet onderscheid wordt gemaakt tussen concepten en individuen. Elk ding zal een rdf:Resource en een individueel ding of een soort ding (rdfs:Class) zijn. Eigenschappen en relaties zullen een rdf:Property zijn:

- tussen individuele dingen;
- tussen soorten dingen;
- tussen een individueel ding en een soort ding;
- tussen individuele dingen en een letterlijke waarde (bijvoorbeeld een tekstwaarde, een numerieke waarde, of een boolean. In het Engels heet dit een 'literal').

Een voorgedefinieerde relatie tussen een individu en een concept is de classificatierelatie waarvoor rdf:type wordt gebruikt. RDFS biedt specialisatierelaties voor concepten (rdfs:subClassOf) en voor attributen/relaties (rdfs:subPropertyOf).

Iedere class in een ontologie kan optioneel direct of indirect worden gedefinieerd als een rdfs:subClassOf van bs:TopConcept. Ieder attribuut in een ontologie kan optioneel direct of indirect worden gedefinieerd als een rdfs:subPropertyOf van bs:topAttribute. Iedere relatie in een ontologie kan optioneel direct of indirect worden gedefinieerd als een rdfs:subPropertyOf van bs:topRelation. Dit is geschematiseerd in figuur 12.



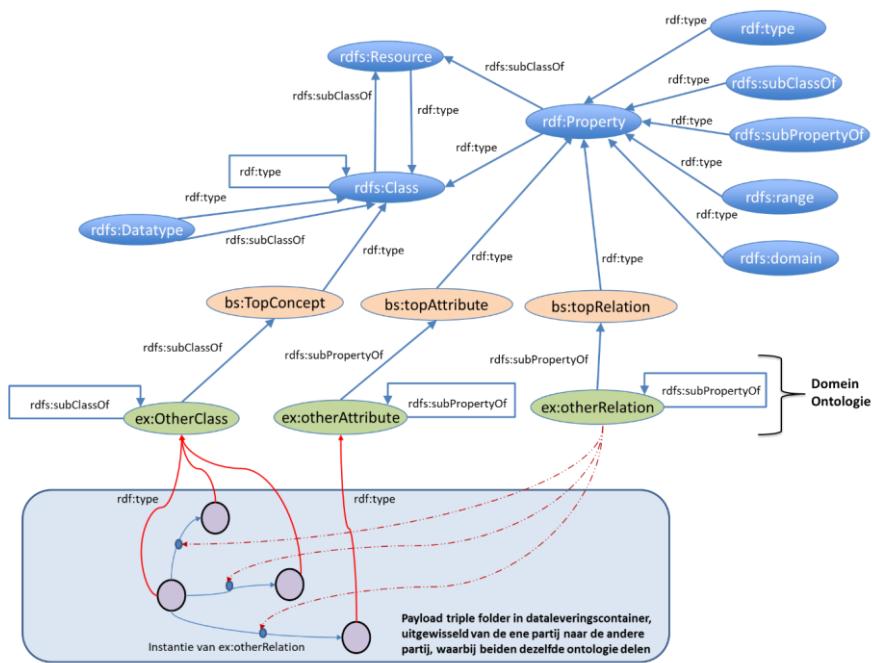
**Figuur 12 — RDFS en verbinding naar ontologie**

In figuur 12 wordt het principe getoond van het uitwisselen van gegevens tussen verschillende partijen, bijvoorbeeld in een project, op basis van overdracht van een 'container' als uitbreiding op figuur 11. Bij het uitwisselen van individuele gegevens die voldoen aan deze NTA, tussen verschillende partijen die dezelfde ontologie hebben, gelden de volgende vereisten:

- Elk individu moet een rdf:type zijn van een concept gedefinieerd in de ontologie.
- Elk attribuut en elke relatie die worden gebruikt, moeten voldoen aan de bijbehorende beperkingen zoals gedefinieerd in de ontologie (vanuit het concept of vanuit het attribuut/relatie gezien).

**OPMERKING** Als voorbeeld 'namespace' wordt in figuur 12 en verder het voorvoegsel (Engels: prefix) 'ex:' gebruikt. De volledige URI zou hier bijvoorbeeld <http://www.example.org/OtherClass> zijn.

## NTA 8035:2020



**Figuur 13 — Uitbreiding van de gedeelde ontologie met individuen**

De exacte RDFS-subset voor deze NTA is opgenomen in bijlage A.

De exacte mapping van het CMM naar RDFS is gegeven in figuur 14:

## NTA 8035:2020

1) Gegevensmodel	>>>	owl:Ontology
2) Gegevensverzameling	>>>	owl:Ontology
3) Concept	>>>	rdfs:Class
• top level-concepten	>>>	instanties van (rdf:type) rdfs:Class
4) Individu	>>>	rdfs:Resource (implicit)
5) Waardetype	>>>	rdfs:Datatype, of rdfs:Class + rdf:Property voor enumeratie datatypen
6) Waarde	>>>	plain of typed literal
7) Attribuut	>>>	
• Annotatie	>>>	rdf:Property
• top level-annotaties	>>>	instanties van (rdf:type) rdf:Property
• Aspect (kwalitatief/kwantitatief)	>>>	rdf:Property
• top level-root 'topAttribute'	>>>	instantie van (rdf:type) rdf:Property (binary & directed)
8) Relatie	>>>	implicit (zelfde file) of rdfs:member
• Groepering	>>>	rdf:type
• Classificatie / Instantiatie	>>>	rdfs:subClassOf & rdfs:subPropertyOf
• Generalisatie / Specialisatie	>>>	niet beschikbaar: te modelleren via bs:hasPart/bs:isPartOf
• Compositie / Decompositie	>>>	(instantie niveau, beperkt op concept niveau) instantie niveau: via triples (subject<-> predicate/object) concept niveau: niet van toepassing (door OWA)
9) Beperking	>>>	rdf:Property
10) Afleiding	>>>	instanties van (rdf:type) rdf:Property
11) Groep	>>>	rdfs:domain & rdfs:range
	>>>	niet van toepassing
	>>>	rdfs:Container

**Figuur 14 — Taalbinding naar RDFS**

Merk op dat we hier owl:Ontology lenen van OWL. RDF/RDFS kent zelf geen groeperingsmechanisme en vaak wordt RDFS bewerkt met RDFS/OWL-software tools die dit ondersteunen.

Een stappenplan voor de ontwikkeling van een RDFS-schema is opgenomen in bijlage E.

SKOS-vocabulaire van het vorige niveau kan worden toegevoegd om de semantiek van RDFS-klassen, enz., te annoteren.

### 7.5 Taalbinding gebaseerd op OWL

Een eerste doel van de extra mogelijkheden van OWL vergeleken met RDFS is om om te gaan met beperkingen. Er zijn class-beperkingen en attribuut/relatie (property)-beperkingen.

In OWL betekent het definiëren van een beperking voor een bepaalde klasse dat die klasse een subklasse wordt van een anonieme 'restrictieklas' waar alle individuen in zitten waarvoor die beperking geldt. Dus als er een klasse Dek is en men wil definiëren dat elk dek ten minste één Plaat als onderdeel heeft, dan wordt er een anonieme superklasse gedefinieerd van het type owl:Restriction die de restrictie kent dat de minimale kardinaliteit van de heeftDeel-relatie gelijk is aan 1 voor een Plaat. Vervolgens wordt Dek een subclass van deze restrictieklasse:

```
ex:Dek
rdf:type owl:Class ;
rdfs:subClassOf [
    rdf:type owl:Restriction ;
    owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
    owl:onClass ex:Plaat ;
```

```
owl:onProperty bs:hasPart ;  
].
```

Er zijn drie soorten class-beperkingen:

1) Value Restrictions: owl:allValuesFrom, owl:someValuesFrom, owl:hasValue

Deze beperkingen zeggen iets over de waarde van een data type property of object property. Bijvoorbeeld dat deze een specifieke waarde/referentie moet hebben (hasValue) of dat, als er een waarde is, deze van een bepaald data type/class moet zijn: allValuesFrom (het logische 'voor alle'), of dat er ten minste een waarde moet zijn van een bepaald data type/class: someValuesFrom (het logische 'er bestaat').

2) Cardinality Restrictions: owl:cardinality, owl:minCardinality, owl:maxCardinality

In plaats van de waarde van een property wordt nu het aantal waarden beperkt. Minimum, maximum of beide tegelijkertijd (min=max).

3) Qualified Cardinality Restrictions (QCR's): owl:qualifiedCardinality, owl:qualifiedMinCardinality, owl:qualifiedMaxCardinality

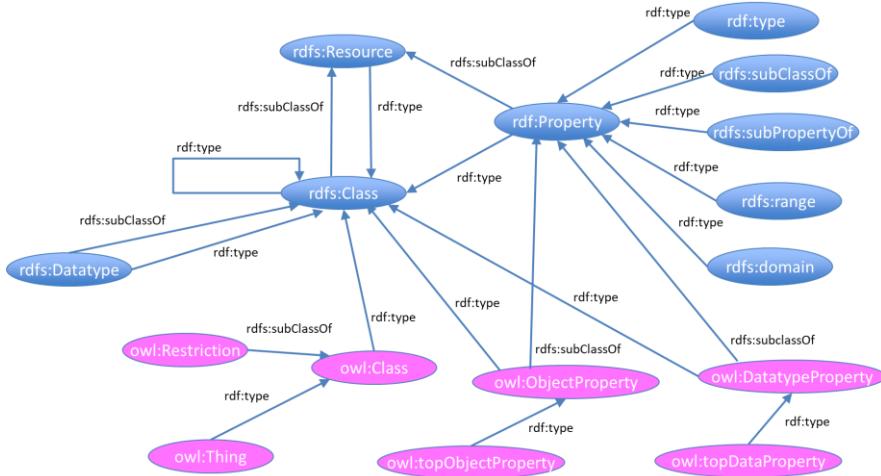
Dit is een variant op 2), waarbij niet alleen het aantal wordt gegeven maar in combinatie met een kwalificatie over het type (net als bij allValuesFrom/someValuesFrom). In plaats van zeggen dat iets drie onderdelen moet hebben, wordt er bijvoorbeeld gezegd dat het twee onderdelen van een bepaald type heeft en een onderdeel van een ander type.

Deze class-beperkingen worden bij een bepaalde class gedefinieerd en werken op een data type property of een object property. Ze beperken het minimum- of maximumaal of zeggen iets over de waarde zelf (voor de property als geheel of specifiek qualified naar een bepaalde target/range).

Een tweede functionaliteit met betrekking tot RDFS is het kunnen maken van onderscheid tussen attributen en relaties door middel van het gebruik van owl:DatatypeProperty respectievelijk owl:ObjectProperty.

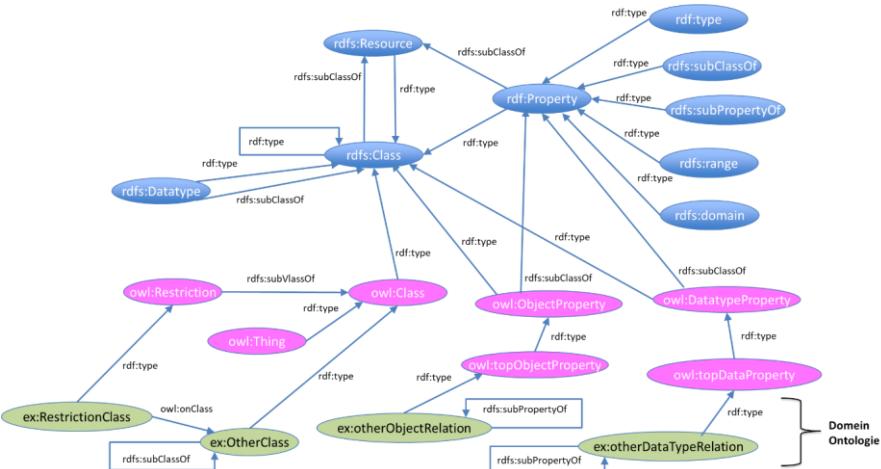
DatatypeProperties hebben betrekking op individuen met lexicale gegevens (bijvoorbeeld strings, getallen, datums, enz.), terwijl ObjectProperties individuen aan andere individuen relateren. Een eigenschap als hasAge is meestal een DatatypeProperty, omdat een leeftijd een getal is, maar een relatie als hasPart is een ObjectProperty, omdat een 'part' (onderdeel) een ander individu zal zijn.

OWL is een uitbreiding op RDFS en is er met OWL meer semantiek beschikbaar gekomen door expliciet data type properties (DatatypeProperty) en object properties (ObjectProperty) te onderscheiden en het kunnen aanbrengen van restricties op zowel klassen (Restriction on Class) als attributen en relaties (Restriction on Property). Een voorbeeld is het kunnen definiëren van een kardinaliteit. In figuur 15 is deze uitbreiding geschematiseerd op basis van figuur 12



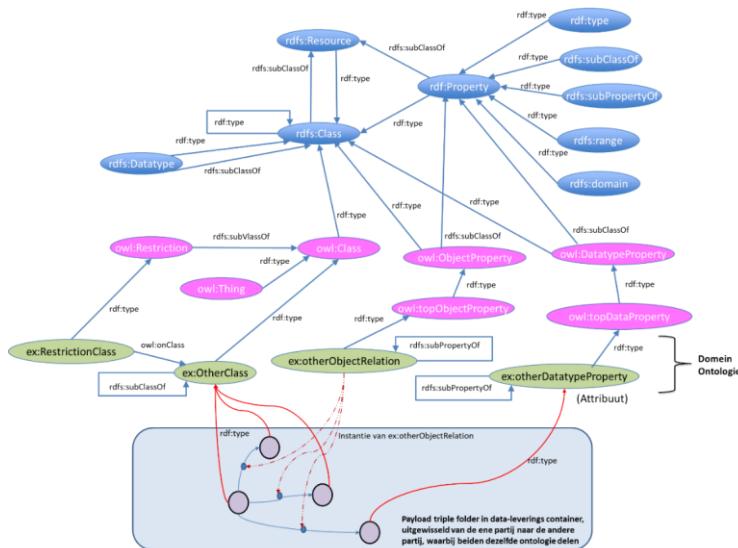
Figuur 15 — OWL als uitbreiding op RDFS

In figuur 16 is op dezelfde manier als in figuur 12 de verbinding getoond naar een ontologie op niveau 2, gebruikmakend van de computerinterpretatiecapaciteiten van OWL.



Figuur 16 — Uitbreiding met ontologieklassen

Op dezelfde manier als voor niveau 1, wordt in figuur 17 de verbinding weergegeven naar de individuen.



Figuur 17 — Uitbreiding met individuen

De exacte OWL-subset voor deze NTA is opgenomen in bijlage A.

De exacte mapping van het CMM naar OWL is gegeven in figuur 18:

1) Gegevensmodel	>>> <code>owl:Ontology</code>
2) Gegevensverzameling	>>> <code>owl:Ontology</code>
3) Concept	>>> <code>owl:Class</code>
• top level-concepten	>>> instances van ( <code>rdf:type</code> ) <code>owl:Class</code> ; <code>TopConcept</code> naar <code>owl:Thing</code>
4) Individu	>>> <code>owl:NamedIndividual</code> of anonymous individual (impliciet)
5) Waardetype	>>> <code>rdfs:Datatype</code> , of <code>owl:Class + owl:ObjectProperty</code> voor enumeratie datatypen; incl. <code>owl:oneOf</code> for gesloten lijsten
6) Waarde	>>> plain of types literal
7) Attribut	>>> <code>owl:AnnotationProperty</code>
• Annotatie	>>> instances van ( <code>rdf:type</code> ) <code>owl:AnnotationProperty</code>
• top level-annotations	>>> <code>owl:DatatypeProperty</code> ; <code>topAttribute</code> naar <code>owl:topDataProperty</code>
• Aspect (kwalitatief/kwantitatief)	>>> (binary & directed)
8) Relatie	>>> impliciet (zelfde file) of <code>rdfs:member</code>
• Groepering	>>> <code>rdf:type</code>
• Classificatie / Instantiatie	>>> <code>rdfs:subClassOf &amp; rdfs:subPropertyOf</code>
• Generalisatie / Specialisatie	>>> niet beschikbaar: te modelleren via <code>bs:hasPart/bs:isPartOf</code>
• Composite / Decompositie	>>> (instantie niveau, beperkt op typeniveau)
• Karakterisatie	>>> instantie niveau: via triples concept niveau: niet van toepassing (door OWA)
• Associate	>>> <code>owl:ObjectProperty</code>
• top level-associates	>>> instances of ( <code>rdf:type</code> ) <code>owl:ObjectProperty</code>
9) Beperking	>>> <code>owl:Restriction (+ details)</code>
10) Afleiding	>>> <code>rdfs:domain &amp; rdfs:range</code>
11) Groep	>>> <code>rdfs:Container</code>

Figuur 18 — Taalbinding naar OWL

## NTA 8035:2020

Een stappenplan voor de ontwikkeling van een OWL-ontologie is opgenomen in bijlage E.

Het SKOS-vocabulaire kan worden gebruikt om de semantiek van OWL-klassen, enz., te annoteren. Ook RDFS-taalconstructies kunnen in OWL worden hergebruikt.

### 7.6 Taalbinding gebaseerd op SHACL

In 6.2.9 van deze NTA werd gesteld dat gegevensuitwisseling of -deling vaak is gebaseerd op de zogenoemde Closed World Assumption (CWA). Dit betekent dat gegevensfeiten die onbekend zijn, als 'niet waar' worden beschouwd, wat zowel de resultaten en wijze van gegevensverificatie als gegevensafleiding beïnvloedt. Het betekent ook dat er binnen zo'n gesloten wereld kan worden vastgehouden aan een Unique Name Assumption (UNA), waarbij de sameAs-functionaliteit wordt vermeden. Standaard houden RDFS en OWL zich echter aan een Open World Assumption (OWA) inclusief verificatie-applicaties en 'reasoners'; dus extra zorg moet worden besteed om ervoor te zorgen dat CWA wordt toegepast (bijvoorbeeld door expliciet 'een schakelaar in te stellen' op de redeneerfunctie).

In plaats van OWL te gebruiken met een niet-standaard interpretatie is het ook mogelijk om een andere semantische webstandaard te kiezen die wel al direct deze interpretatie heeft: SHACL. Elke OWL-restrictie heeft een SHACL shape-tegenhanger. Vele kunnen zelfs automatisch worden afgeleid. Bovendien kan SHACL worden gecombineerd met RDFS, of zelfs met OWL in het geval dat ook een OWA-interpretatie handig is voor andere doeleinden dan louter een gegevenslevering (bijvoorbeeld informatie-inwinning).

SHACL heeft meer voordelen ten opzichte van OWL dan alleen de CWA-interpretatie waarmee deze NTA werkt:

- SHACL biedt één standaard CWA-interpretatie (geen afhankelijkheid van een specifieke redeneer-/verificatie-applicatie).

OPMERKING 1 De CWA-interpretatie bij OWL kan software-afhankelijk zijn.

- SHACL kan omgaan met beperkingen op het niveau van een individu (een of een groep) via zogenoemde 'custom targets'.

OPMERKING 2 Dit is een heel belangrijke functie voor het vastleggen van specificerende beperkingen.

- SHACL is krachtiger en flexibeler dan OWL:

- Het geeft ondersteuning van logische uitdrukkingen (Engels: logical expressions) van beperkingen.

- Het biedt een vergelijking van gegevenswaarden.

- Elk type beperking gaat uiteindelijk via SPARQL.

- SHACL is flexibeler in hergebruik van generieke ontologieën:

- Het biedt de mogelijkheid om shapes te deactiveren (bijvoorbeeld die zijn verkregen door overerving van superklassen).

- SHACL zal in de toekomst regels voor afleiding/inferentie ondersteunen via de Advanced Features (AF)-specificatie (nu nog in draft).

— SHACL kan software-implementaties helpen, zoals bijvoorbeeld het declaratief vastleggen van een Grafische User Interface (GUI).

VOORBEELD <https://www.w3.org/TR/shacl/#nonValidation>.

*Voorbeeld van standaard SHACL*

Stel dat in OWL de volgende beperking bestaat:

```
ex1:Bridge
rdfs:subClassOf [
    rdf:type owl:Restriction ;
    owl:cardinality "1"^^xsd:nonNegativeInteger ;
    owl:onProperty ex1:height ;
]
```

die zegt dat een brug exact één hoogtewaarde heeft. Dit kan worden omgezet in een SHACL shape:

```
ex1:Bridge
sh:property [
    sh:path ex1:height ;
    sh:minCount 1 ;
    sh:maxCount 1 ;
]
```

*Voorbeeld van Advanced SHACL: het modelleren van afleidingen*

Het betreft de volgende SHACL shape:

```
bb:Toilet
sh:rule [
    rdf:type sh:SPARQLRule ;
    sh:construct """
        CONSTRUCT {
            $this bb:area ?area .
        }
        WHERE {
            $this bb:width ?width .
            $this bb:depth ?depth .
    
```

## NTA 8035:2020

```
BIND (?width * ?depth AS ?area) .  
}  
;  
sh:message "Calculation of toilet area" ;  
sh:prefixes <https://w3id.org/bouwbesluit> ;  
].
```

Deze shape definieert dat de waarde voor een gebied (voor elke instantie van bb:Toilet) wordt berekend als de vermenigvuldiging van de breedte en diepte, met behulp van de standaard SPARQL query-taal (een construct-query om precies te zijn).

Beperkingen en afleidingen kunnen hierbij flexibel worden gemixt.

De exacte SHACL-subset voor deze NTA is opgenomen in bijlage A.

De exacte mapping van het CMM naar SHACL is gegeven in figuur 19:

1) Gegevensmodel	>>>	owl:Ontology
2) Gegevensverzameling	>>>	owl:Ontology
3) Concept	>>>	sh:NodeShape
• Top level-concepten	>>>	instanties van (rdf:type) sh:NodeShape
4) Individu	>>>	owl:NamedIndividual of anonymous individual (impliciet)
5) Waardetype	>>>	rdfs:Datatype, of owl:Class + owl:ObjectProperty voor enumeratie data typen
6) Waarde	>>>	plain of typed literal (impliciet)
7) Attribuut	>>>	
• Annotatie	>>>	owl:AnnotationProperty
• top-level annotaties	>>>	instanties van (rdf:type) owl:AnnotationProperty
• Aspect (kwalitatief/kwantitatief)	>>>	owl:DatatypeProperty & sh:PropertyShape (binary & directed)
8) Relatie	>>>	impliciet (zelfde file) of rdfs:member rdf:type
• Groepering	>>>	rdfs:subClassOf & rdfs:subPropertyOf
• Classificatie / Instantiatie	>>>	niet beschikbaar: te modelleren via bs:hasPart/bs:isPartOf (instantie niveau, beperkt op type niveau)
• Generalisatie / Specialisatie	>>>	
• Compositie / Decompositie	>>>	
• Karakterisatie	>>>	instantie niveau: via triples concept niveau: via sh:property
• Associatie	>>>	owl:ObjectProperty & sh:PropertyShape
• top level-associaties	>>>	instanties van (rdf:type) owl:ObjectProperty & sh:PropertyShape
9) Beperking	>>>	sh:property, sh:in, sh:PropertyShape (+details)
10) Afleiding	>>>	sh:Rule (+details)
11) Groep	>>>	rdfs:Container

**Figuur 19 — Taalbinding naar SHACL**

Een stappenplan voor de ontwikkeling van een SHACL-ontologie is opgenomen in bijlage E.

Ook nu kan SKOS/RDFS-vocabulaire worden gebruikt om de semantiek van shapes te annoteren. In het bijzonder kunnen RDFS- en OWL-taalconstructies (anders dan annotaties) in SHACL worden hergebruikt.

## 7.7 Identificatie, naamgeving en annotatie

### 7.7.1 Identificatie: URI-strategie

De URI-strategie in deze NTA is gebaseerd op de volgende bronnen:

- CEDR Interlink URI Strategie,  
[https://www.roadotl.eu/static/media/INTERLINK\\_D4\\_Defining\\_the\\_Principles\\_90kqubw.PDF](https://www.roadotl.eu/static/media/INTERLINK_D4_Defining_the_Principles_90kqubw.PDF),  
vanaf p. 84;
- [https://www.geonovum.nl/uploads/documents/D1-2013-09-19\\_Towards\\_a\\_NL\\_URI\\_Strategy.pdf](https://www.geonovum.nl/uploads/documents/D1-2013-09-19_Towards_a_NL_URI_Strategy.pdf);
- Best Practices for Publishing Linked Data, W3C Working Group Note 09 January 2014,  
<https://www.w3.org/TR/lid-bp/>;
- Cool URIs for the Semantic Web, W3C Interest Group Note 03 December 2008,  
<https://www.w3.org/TR/cooluris/>.

De URI heeft de algemene vorm:

```
uri := http[s]://{internet domain}/{path}/{x}, where
      — {x} := def/[data model](/#)[({concept}|{attribute}|{relation})];
      — {x} := id/[data set](/#)[individual];
      — {x} := doc/{document name}[document extension].
```

waarin:

**def** is om een gegevensmodel, gegevensverzameling of (hierbinnen) een concept, attribuut of relatie aan te geven;

**id** is om naar iets aanwijsbaar (een "individu") in de echte wereld te verwijzen;

**doc** is om documentatie over een gegevensmodel, gegevensverzameling, concept, attribuut, relatie of individu aan te geven.

Meta-symbolen:

:=	is gedefinieerd door ('productieregel')
{ ... }	verplichte component
[ ... ]	optionele component
( ... )	groepering
	exclusief OF

Een belangrijk verschil met INTERLINK is dat er geen versie-informatie in de URI wordt aanbevolen voor datamodelen en dataverzamelingen omdat die zo stabiel en niet-semantisch mogelijk moet zijn. Als hier toch behoefte aan is, dan is het aan te bevelen de Interlink-benadering te volgen (versieloze varianten en versievarianten waarbij de versieloze variant altijd naar de laatste versie verwijst):

## NTA 8035:2020

uri := http[s]://{{internet domain}}/[path]/[version]/{x}.

Internet services die stabiele URI's bieden, zoals w3id.org of purl.org, kunnen worden gebruikt.

### VOORBEELDEN

- <http://www.interlink.eu/def/roadotl#Road> ;
- <http://www.roadnetwork.nl/id/nwb#A16> ;
- <http://www.interlink.eu/doc/RoadDefinition.docx> ;
- [http://www.roadnetwork.nl/id/traffic-control/light\\_3456-c](http://www.roadnetwork.nl/id/traffic-control/light_3456-c) ;
- <https://w3id.org/def/basicsemantics-owl#> .

OPMERKING 1 Data model en data set zijn optionele fragmenten van de URI, omdat deze soms al in het domein en/of pad zijn verwerkt.

OPMERKING 2 De keuze tussen '/' of '#' hangt samen met de verwachte grootte van de datamodellen/datasets (als groot: gebruik '/', meer achtergrondinformatie: <https://www.w3.org/TR/cooluris/#hashuri>).

OPMERKING 3 Bij voorkeur zijn de URI's niet alleen voor identificatie maar ook voor lokalisatie (maak datastructuren en data 'dereferenceable').

OPMERKING 4 Extensies in namen van gegevensmodellen en gegevensverzamelingen worden niet gebruikt. Gebruik standaard 'server negotiation' voor selectie van de juiste representatie (inclusief mensleesbare introducties als .html).

### 7.7.2 Naamgeving en annotatie

Met betrekking tot de lokale namen voor het concept of het referentiefragment van de URI's worden gebruikersvriendelijke namen aanbevolen, geen codes. Als er codes worden gebruikt, zorg er dan voor dat in elk geval door mensen leesbare labels worden gebruikt in de vorm van RDFS- of SKOS-labels.

Bij voorkeur worden (precies 1) skos:prefLabel en (0 of meer) skos:altLabels gebruikt. De laatste kunnen worden gebruikt voor synoniemen. Het is ook mogelijk om skos:altLabel voor afkortingen te gebruiken, maar dit verdient geen aanbeveling, omdat een synoniem en een afkorting verschillende dingen zijn. Voor afkortingen worden 0 of meer specifieke annotatie(s) gehanteerd:

- bs:abbreviation

*Voor concepten, attributen, relaties en waardetypen*

Bij het gebruik van namen:

- Namen van Concepten en Waardetypen beginnen altijd met een hoofdletter (CamelCase).
- Namen van attributen en relaties beginnen altijd met een kleine letter (camelCase).
- Namens zijn bij voorkeur in het Engels en in enkelvoudsvorm.

*Voor individuen*

Dit is flexibel. Er zijn de volgende opties:

- specifieke namen, zoals A16 voor een Nederlandse weg;
- genummerde instanties voor een klasse, zoals Road\_16;
- semantiek-vrije, gegenereerde UUID's, zoals 550e8400-e29b-41d4-a716-446655440000.

Termdefinities worden vastgelegd met hergebruik van het Simple Knowledge Organization System (SKOS). Hierbij is nadrukkelijk ook gekeken naar bestaande specificaties zoals BP4mc2, Best Practices for meaningful connected computing ([20]).

- skos:definition.

Dit geeft een speciekkere betekenis dan het algemenere rdfs:comment.

- rdfs:isDefinedBy

Dit kan worden gebruikt om naar een externe bron te verwijzen die de onderwerpresource definiert. Deze eigenschap kan worden gebruikt om een RDF-vocabulaire aan te duiden waarin een bron wordt beschreven, of de definitie te beschrijven van de onderwerp-resource. Het verwijst naar de informatie over de bron (die niet noodzakelijkerwijs in RDF is beschreven).

- rdfs:seeAlso

Dit is een generiekere variant die typisch wordt gebruikt voor niet-RDF-bronnen.

Codes (onafhankelijk van elke taal) worden opnieuw gedefinieerd met behulp van SKOS:

- skos:notation

Elk van de gebruikte codes/notaties moet op unieke wijze de dingen identificeren die de code/notatie hebben.

Tabel 1 geeft een overzicht van de voor deze NTA gekozen annotatietypen.

**Tabel 1 — Gekozen annotatietypen**

	<b>In taalbinding</b>
label	rdfs:label of skos:prefLabel (gedetermineerd)
synoniem	skos:altLabel
code	skos:notation
opmerking van auteur	skos:editorialNote
scope	skos:scopeNote
commentaar	rdfs:comment
externe informatie	rdfs:seeAlso
voorbeeld	skos:example
is intern gedefinieerd door	skos:definition

	<b>In taalbinding</b>
is extern gedefinieerd door	rdfs:isDefinedBy
afkorting	bs:abbreviation

**OPMERKING** IRI werd in 2005 door de Internet Engineering Task Force (IETF) gedefinieerd als een nieuwe internetstandaard om het bestaande URI-schema uit te breiden. De primaire standaard wordt gedefinieerd door IETF RFC 3987. Hoewel URI's beperkt zijn tot een subset van de ASCII-tekenset, kunnen IRI's tekens bevatten uit de universele tekenset (ISO/IEC 10646), inclusief Chinese of Japanse kanji, Koreaanse, Cyrilische tekens, enz.

Specifieke HTTP URI's worden gebruikt, zodat deze dingen kunnen worden doorverwezen en opgezocht ('dereferenced') door mensen of software.

**VOORBEELD** Een voorbeeld van een Universally Unique Identifier (UUID) volgens ISO/IEC 11578 is:  
<http://example.com/myproject#40bb99c0-1f74-11e2-81c1-0800200c9a66>.

## 7.8 Enumeratiедatotypen

Enumeratiедatotypen worden gemodelleerd als klassen/shapes, zodat toegelaten opsommingsitems individuen met meertalige labels en beschrijvingen of definities, enz., kunnen worden. Ze worden een subclass van bs:EnumerationType zodat ze onder zelfde noemer vallen en makkelijk te ondervragen zijn. Er wordt onderscheid gemaakt tussen open/uitbreidbare en gesloten/gefixeerde enumeratiедatotypen. De gesloten varianten kunnen worden gedefinieerd met behulp van de 'open world' owl:oneOf of de 'closed world' sh:in.

In sommige opsommingen zit een ordening ([Low, Medium, High]), andere zijn eigenlijk platgeslagen (niet-uitgemodelleerde) klassen ([Tuibrug, Hangbrug]).

**VOORBEELD** Een voorbeeldbestand voor eenvoudige attributen is opgenomen in bijlage F. Dit is een fragment voor een enumeratie met een ordening:

```

ex1:LoadLevelType
  a owl:Class ;
  rdfs:subClassOf bs:EnumerationType ;
  owl:equivalentClass [
    a owl:Class ;
    owl:oneOf (
      ex1:Light
      ex1:Normal
      ex1:Heavy
    ) ;
  ] ;
.

```

Er wordt een OWL-kLASSE, genaamd LoadLevelType gedefinieerd. Deze OWL-kLASSE is 'gesloten' (er kan geen ander niveau worden gebruikt zoals bijvoorbeeld ex1:VeryHeavy) door de owl:oneOf-constructie, waarbij alle

toegelaten items zijn gedefinieerd als individuen van deze klasse. Vervolgens wordt een owl:ObjectProperty gedefinieerd met deze datatypeklasse als range:

```
ex1:loadLevel rdf:type owl:ObjectProperty .
```

Deze object property loadLevel kan worden gebruikt door individuen van de voertuigklasse (mogelijk verder beperkt op voertuigklasseniveau):

```
ex1:Vehicle_1 rdf:type ex1:Vehicle ;
    ex1:loadLevel ex1:Heavy .
```

OPMERKING 1    Enumeratie-items kunnen lid zijn van meerdere enumeratiedatatypes.

OPMERKING 2    Enumeratie-items kunnen ook uitgewerkt voorkomen in dezelfde ontologie. Dan is er echter wel sprake van reificatie en dus "OWL Full":

Brug\_123 rdf:type TuiBrug rdf:type BrugType

De verwachting is echter dat deze situatie niet snel zal voorkomen binnen één ontologie (binnen één view).

Als het aantal toegelaten enumeratie-items groot wordt,<sup>10</sup> kan er voor worden gekozen om de klassen te subclassen met bijbehorende subproperties. Ook kan door middel van standaard restricties [per klasse](#) een beperkte keuze uit een grote verzamelingen worden gemaakt [per klasse](#).

## 7.9 Decompositie

In de praktijk bestaan er vele interpretaties voor decompositie. Bijvoorbeeld: soms op een transitieve manier, soms op een niet-transitieve manier. Dit abstractiemechanisme is ook speciaal, omdat het niet direct wordt ondersteund door de talen die worden gebruikt.

In deze NTA wordt een niet-transitieve bs:hasPart gedefinieerd. Zijn inverse, de isPartOf relatie, wordt niet expliciet gemodelleerd<sup>10</sup>. De transitieve variant kan worden afgeleid wanneer dat nodig is. Verder gaat het om een sterke variant (vergelijk UML's compositie). Dat wil zeggen dat wanneer een geheel ophoudt te bestaan, zijn delen ook ophouden te bestaan.

In het geval dat meerdere manieren van decompositie relevant zijn, wordt aanbevolen om aparte ontologieën te ontwikkelen die elk een specifieke decompositie vastleggen (één decompositie per view).

## 7.10 Grootheden en eenheden

Grootheden ('quantity kinds') kunnen worden beschouwd als kwantitatieve referentieattributen, eenheden ('units') als referentiewaarden voor deze attributen. Er zijn vele systemen om grootheden en eenheden semantisch te modelleren. Deze NTA hanteert de QUDT-ontologie [23], ontwikkeld door NASA en TopQuadrant. QUDT is volledig afgestemd met ISO/IEC 80000 (systematiek, namen, definities, symbolen, enz.). De laatste versie 2.1 is nog volop in ontwikkeling maar het deel wat in deze NTA wordt gebruikt, is stabiel en beschikbaar.

In deze NTA worden eigenschappen standaard op de meest directe en eenvoudige manier gemodelleerd, zoals als owl:DatatypeProperty's bij het toepassen van LoC-3a.

### VOORBEELD

ex:height a owl:DatatypeProperty ;

<sup>10</sup> RDF triples kunnen twee kanten op doorlopen worden. Explicet definiëren van de inverse relatie is alleen nodig ten behoeve van eventuele inverse beperkingen die hier niet van toepassing zijn.

## NTA 8035:2020

```
bs:unit unit:M ;
bs:quantityKind quantitykind:Length .
```

In de complexe variant kunnen waarden en eenheden (zie voor meer informatie par. 8.2 en bijlage G) op dezelfde manier worden bevestigd aan instanties van hoogte.

### VOORBEELD

```
ex:Height_1 a bsc:Property ;
  rdf:value 12.4 ;
  bs:unit unit:M ; -- optioneel, nu is het mogelijk om een 'eenheid per waarde' te specificeren ;
  bsc:hasPropertyDef ex:height .
```

## 7.11 Implementatie van het Conceptueel Meta Model

Het CMM is in deze editie van de NTA geïmplementeerd in een OWL-ontologie volgens de LoC-3a OWL-taalbinding. In een volgende NTA-editie zullen ook de andere taalvarianten een dergelijke implementatie krijgen. OWL wordt nu gezien als meest relevant, gegeven de geplande use case-typen voor de OTL'en.

Deze implementatie wordt in deze NTA basicsemantics-owl genoemd en is opgenomen in bijlage D.

Deze ontologie kan in elke andere eindgebruikerontologie worden geïmporteerd, zodat alle afgesproken constructies direct voor hergebruik ter beschikking staan.

In bijlage E worden in detail de stappen gegeven voor de opbouw of verbouw van nieuwe respectievelijk bestaande gegevensstructuren afhankelijk van het gewenste modelleerniveau.

In de praktijk hoeft er niet per se een keuze voor één LoC te worden gemaakt. Eén of meerder LoC's kunnen ook parallel worden gehanteerd. Deze kunnen worden geïntegreerd:

```
ex:Bridge rdf:type skos:Concept, owl:Class, sh:NodeShape ; ...
```

Maar vaak is het beter deze LoC-views separaat te modelleren en te relateren:

in een skos namespace:

```
— skos-ex:Bridge rdf:type skos:Concept .
```

in een owl namespace:

```
— owl-ex:Bridge rdf:type owl:Class ;
```

```
— skos:related skos-ex:Bridge .
```

In plaats van skos:related kan ook rdfs:seeAlso worden gebruikt.<sup>11</sup>

<sup>11</sup> Hiervoor kan ook wel dct:subject worden gebruikt, of sh:shapesGraph/sh:suggestedShapesGraph om vanuit RDFS/OWL of data files naar SHACL graphs te verwijzen.

## 8 Conceptueel Model (CM)

### 8.1.1 Top Level-concepten

#### 8.1.1.1 Inleiding

Deze NTA hanteert een Conceptueel Top Level Model (CM) als instantie van (een deel van) het Conceptueel Meta Model (CMM) uit hoofdstuk 6. Omdat in deze NTA RDF-gebaseerde taalbindingen worden geboden, is dit CMM direct ook een gegevensmodel (data model).

Eerst wordt een topniveau gedefinieerd van zeven concepten op hetzelfde (specialisatie)niveau die instanties zijn van het metaconcept Concept onder een root TopConcept:

— TopConcept:

- 1) FysiekObject;
- 2) InformatieObject;
- 3) Activiteit;
- 4) Toestand;
- 5) Gebeurtenis;
- 6) RuimtelijkGebied;
- 7) TemporeelGebied.

Deze disjuncte basisconcepten kunnen worden gebruikt om een taxonomie en/of meronomie op te bouwen die weer als ruggengraat kan/kunnen dienen voor een ontologie.

Er wordt onderscheid gemaakt in objecten, onderverdeeld in fysieke objecten en informatie-objecten, die 'zijn' en activiteiten die 'plaatsvinden'. Het metamodel introduceert als taal nog geen 'tijdaspecten'. Dit moet worden geregeld door de modellering in de taal. Dat gebeurt door de introductie van dynamische concepten voorbij de statische objecten en activiteiten. Toestanden van objecten en activiteiten en gebeurtenissen, waarbij toestanden beginnen en eindigen in de tijd en deze daarmee dynamisch maken. Wanneer tijd niet relevant is omdat er slechts tijdroze statische aspecten hoeven te worden gemodelleerd of momentopnamen ('snapshots' of 'baselines') van objecten/activiteiten in de tijd, dan kunnen activiteiten, toestanden en gebeurtenissen worden genegeerd.

**OPMERKING** Tijdroos modelleren wordt vaak de 3D-aanpak genoemd. Wanneer meerdere momentopnamen in hetzelfde model gemodelleerd moeten worden, is zowel het toestand- als het gebeurtenisconcept nodig. Dit wordt vaak de 4D-aanpak genoemd.

In 7.12.1.2 t/m 7.12.1.8 worden de zeven concepten stuk voor stuk toegelicht. Tussen haakjes wordt de Engelse term weergegeven.

#### 8.1.1.2 FysiekObject (PhysicalObject)

Een FysiekObject is iets dat bestaat of zou kunnen bestaan in ruimte en tijd, een manifestatie en een afbakening van materie en/of energie vormt en waarneembaar is door de zintuigen. Een fysiek object voert activiteiten uit en wordt ook getransformeerd door activiteiten.

**VOORBEELD** Een viaduct, een lichtmast, een pomp, een auto.

OPMERKING 1 Een zelfde fysiek object kan zowel virtueel bestaan, bijvoorbeeld op een tekening of in de vorm van een digitaal model, en in de werkelijke wereld (geconstrueerd en in gebruik). Het betreft hier de verschillende levenscycli waarin een object kan verkeren.

OPMERKING 2 [ISO 15926] kent een verdere specialisatie van een fysiek object naar Functioneel Fysiek Object en een Gematerialiseerd Fysiek Object. Het Functioneel Fysiek Object is bedoeld om een gespecificeerd ontwerp inclusief functies van een fysiek object te representeren. Het Gematerialiseerd Fysiek Object geeft in de 'echte wereld' invulling aan c.q. is geïnstalleerd als het Functioneel Fysiek Object. Het Gematerialiseerd Fysiek Object zal ten gevolge van slijtage, degeneratie of falen vervangen worden door een nieuw Gematerialiseerd Fysiek Object, waarbij ook deze vervanging voldoet aan de specificaties van zijn tegenhanger, het Functioneel Fysiek Object. Als voorbeeld is de pomp met code P-101 op een processchema een Functioneel Fysiek Object en de pomp die hiervoor gekocht en geïnstalleerd is met serienummer AW30456 een Gematerialiseerd Fysiek Object, waarbij beide geklassificeerd zijn als een Pomp op basis van bijvoorbeeld een ontologie. Zo kan ook een Brug als opgenomen in een tracébesluit, worden beschouwd als een Functioneel Fysiek Object dat uiteindelijk wordt gerealiseerd respectievelijk gematerialiseerd als een daadwerkelijke Brug met bijvoorbeeld laaggelegen landhoofden en een brugdek van voorgespannen betonelementen.

OPMERKING 3 Ook een (levend) organisme is een Fysiek Object, waarmee dus ook een mens een Fysiek Object is. Automatisch is een organisatie van mensen ook een Fysiek Object.

#### 8.1.1.3 InformatieObject (InformationObject)

Een InformatieObject is een op zichzelf staand geheel van gegevens met een eigen identiteit.

OPMERKING 1 Een informatie object betreft een abstract concept dat met behulp van een fysiek object (als 'drager') een door zintuigen waarneembare vorm kan aannemen.

OPMERKING 2 NEN 2082 geeft als voorbeeld van een informatie object: document, databasegegeven, e-mailbericht (met bijlagen), (zaak)dossier, internetsite (of een deel ervan), foto/afbeelding, geluidopname, wiki, blog, enz. Deze voorbeelden kunnen hiermee worden geïnterpreteerd als zowel een instantie van een informatie object als een instantie van een fysiek object.

OPMERKING 3 De Nederlandse Overheid Referentie Architectuur (NORA) definieert Gegevens als de weergave van een feit, begrip of aanwijzing, geschikt voor overdracht (uitwisseling of deling), interpretatie of verwerking door een persoon of apparaat. Het betreft hier alle vormen van gegevens, zowel data uit informatiesystemen als 'records' en documenten, in alle vormen, zowel gestructureerd als ongestructureerd. Ook deze voorbeelden kunnen hiermee worden geïnterpreteerd als zowel een instantie van een informatie object als een instantie van een fysiek object.

#### 8.1.1.4 Activiteit (Activity)

Een Activiteit is iets dat plaatsvindt of zou kunnen plaatsvinden in een concrete dan wel virtuele ruimte of tijd. Een activiteit transformeert fysieke objecten en/of informatie objecten en wordt uitgevoerd door een fysiek object. Informatie objecten kunnen als input of sturing dienen voor het uitvoeren van een activiteit.

OPMERKING 1 Volgens de systeemkunde (zie bijvoorbeeld [4]) omvat een proces een serie transformaties tijdens de doorvoer van een of meer fysieke objecten en/of informatie objecten. Een proces kan binnen deze context worden beschouwd als een set van samenhangende activiteiten om input (bijvoorbeeld planningen, specificaties, adviezen) om te zetten in output (bijvoorbeeld inspecties, onderzoeksrapporten, handhavingsmaatregelen).

OPMERKING 2 De functie van een systeemelement is de activiteit die het uitvoert, zodanig dat de output van die activiteit bijdraagt aan het doel dat de betrokken stakeholder wil bereiken.

Het betreft de bijdrage die van de medewerkers respectievelijk het voorwerp wordt verwacht voor het behalen van de ondernehmingsresultaten respectievelijk de systeemprestatie waar het voorwerp onderdeel van uitmaakt.

**Met opmerkingen [wb7]:** Zie commentaar 8.2

Is dit ook:

NVN-ISO/TS 15926-11, Industrial automation systems and integration - Integration of life-cycle data for process plants including oil and gas production facilities - Part 11: Methodology for simplified industrial usage of reference data

?

Ik kan ISO 15926-11 verder niet vinden.  
Graag opnemen in h 2. (En schrappen in bibliografie.) Deze norm wordt immers ook normatief gebruikt in 8.2.

**Met opmerkingen [wb8R7]:** Graag bevestigen.

**Met opmerkingen [BH(9R7):** In laatste overleg, leo

**Met opmerkingen [BH(10R7):** ja dat is hem! gechecked met leo, verwerken jullie?

Ook een functie (zeker mathematische) kent een input en een output, analoog aan activiteiten.

Over het algemeen is iets een functie als dezelfde bijdrage kan worden geleverd met verschillende middelen. Het leveren van stroom (=functie) kan met behulp van een batterij (=taak van dit FysiekObject), maar ook met een aggregaat (=taak van dit FysiekObject).

OPMERKING 3 Op basis van opmerkingen 1 en 2 zijn proces en functie te zien als specialisatie van Activiteit.

#### 8.1.1.5 Toestand (State)

Toestand is het geheel van omstandigheden of condities waarin iets of iemand zich bevindt op een specifiek moment in de tijd.

De toestand van een systeem op een bepaald moment bevat de waarden van de eigenschappen in het systeem op dat moment. Als de waarde van een eigenschap van een element verandert, verandert de toestand van het systeem. Deze verandering vindt plaats door een bepaalde gebeurtenis of een activiteit.

OPMERKING 1 De toestand is niet altijd een unieke verzameling eigenschapswaarden: er kunnen immers verschillende verzamelingen zijn die elk een eigen toestand aanduiden. Er kunnen dus meerdere toestanden tegelijk gelden voor een enkel ding, elk beschouwd vanuit een specifieke context. Een auto kan zich zowel in de toestand 'Rijden' bevinden als in de toestand 'Onderhoud benodigd'.

OPMERKING 2 'Calamiteit', 'Onderhoud' en 'Normaal' zijn gedefinieerde toestanden van een tunnelsysteem. Onder het aanwezig zijn van voorwaarden (gebeurtenissen) gaat de ene toestand over in de andere toestand. In een ontologie zullen gedefinieerde toestanden in de taxonomie moeten worden opgenomen.

OPMERKING 3 NEN 2767: reeks kwalificaties van de toestand of conditie van gebouw en gebouwgebonden installaties: 1) Uitstekende conditie; 2) Goede conditie; 3) Redelijke conditie; 4) Matige conditie; 5) Slechte conditie; 6) Zeer slechte conditie.

#### 8.1.1.6 Gebeurtenis (Event)

Een Gebeurtenis is een overgang in toestand (van een object of een activiteit).

OPMERKING 1 In de gewone betekenis is een gebeurtenis een opmerkelijk voorval, zoals een glas dat stukvalt, of een ontmoeting tussen twee personen. In het eenvoudigste geval en bij benadering vindt een gebeurtenis plaats op één plaats en één tijd. Dit is een idealisering, een echte gebeurtenis heeft zowel in ruimte als in tijd enige uitgestrektheid.

OPMERKING 2 Het bereiken van een mijlpaal is een significante gebeurtenis in een project en/of werkpakket.

#### 8.1.1.7 RuimtelijkGebied (SpatialRegion)

Met opmerkingen [BH(11): aangepast]

Een RuimtelijkGebied is een afbakening in een concrete of abstracte ruimte.

OPMERKING 1 Het gaat hier primair om een topologische/geometrische ruimte. In een topologische ruimte is het mogelijk om te spreken over de topologische relaties (omvat/bevindt zich binnen, bevindt zich buiten, grenst aan) tussen ruimtelijke gebieden en daarmee indirect ook tussen twee objecten of activiteiten gerelateerd aan die ruimten door inname of begrenzing. In een geometrische ruimte (per definitie ook een topologische ruimte) is het mogelijk om daarnaast ook afstanden te benoemen tussen ruimten cq. objecten of activiteiten en is het daarmee mogelijk om de locatie van (een kenmerkend geometrisch element van) een object of activiteit te specificeren ten opzichte van een verzameling van referentie-objecten/activiteiten. Daarbij wordt de ruimtelocatie vaak uitgedrukt in een ruimtereferentie (een tupel, die een set van gegeneraliseerde coördinaten vormt) en de aanduiding van het gebruikte ruimtereferentiesysteem.

## NTA 8035:2020

OPMERKING 2 De hiervoor genoemde topologische relaties maken geen deel uit van dit Conceptueel Top Level Model, maar kunnen worden opgenomen in een domein-specifieke ontologie, of hergebruikt uit bestaande ontologieën.

OPMERKING 3 De concrete ruimte kent een onderverdeling naar 0D (punt), 1D (lijn), 2D (vlak) en 3D (ruimtelijk lichaam).

OPMERKING 4 In het spoordomein is het Profiel van Vrije Ruimte (PVR) de ruimte boven en naast een spoor waarbinnen zich geen vaste of tijdelijke voorwerpen mogen bevinden, die het railvervoer zouden kunnen hinderen en/of in gevaar brengen.

OPMERKING 5 Indien men de fysieke eigenschappen van (bijvoorbeeld) een vergaderruimte wil beschrijven (luchtkwaliteit, temperatuur, hoeveelheid licht etc.) dan kan de vergaderruimte worden gemodelleerd als (een subklasse van) een fysiek object. Het gaat dan om de eigenschappen van het fysieke gaslichaam, bestaand uit atmosferische lucht, dat wordt begrensd, of zelfs vastgehouden, door het fysieke bouwwerk, waarin de vergaderruimte zich bevindt. In een domein-specifieke ontologie kan een taxonomie van dit soort "fysieke ruimtes" worden opgenomen.

OPMERKING 6 Indien men de locatie van een fysiek object of activiteit wil uitdrukken m.b.v. een adres of een set van coördinaten, dan kan een relatie heeftLocatieReferentie worden geïntroduceerd. Deze relatie drukt de locatie uit in een combinatie van een tupel van getallen en een locatie referentie systeem (LRS), waarbinnen het tupel kan worden geïnterpreteerd. Voorbeelden van locatie referentie systemen zijn een lokaal cartesisch coördinatensysteem in een kamer, adresseringssystemen (postcode, BAG), lineaire referentiesystemen (BPS) en georeferentiesystemen (WGS84, ETRS89, RDNAP).

In het conceptuele model in bijlage D zijn de ook de basis modelleerconstructies opgenomen om de locatie in de ruimte vast te leggen:

- `bs:hasSpatialLocation`
- `bs:hasSpatialReferenceSystem`
- `bs:hasSpatialReference`
- `bs:SpatialReferenceSystem` (en diverse specialisaties)
- `bs:SpatialReference`

### 8.1.1.8 TemporeelGebied (TemporalRegion)

Met opmerkingen [BH(12): aangepast!]

Een TemporeelGebied is een afbakening of interval in de tijd, begrensd door twee tijdstippen of gebeurtenissen.

VOORBEELD Een arbeidsovereenkomst kan worden aangegaan voor een bepaalde tijd. Die tijd kan in een periode zijn vastgelegd of gekoppeld aan gebeurtenissen (bijvoorbeeld begin en einde van een project).

OPMERKING Faseren is het opdelen van het werk in afzonderlijke tijdsperioden (bijvoorbeeld projectdefinitiefase en projectuitvoeringsfase), elk met zijn eigen vooraf gedefinieerde resultaat.

In het conceptuele model in bijlage D zijn de ook de basis modelleerconstructies opgenomen om de locatie in de tijd vast te leggen:

- `bs:hasTemporalLocation`
- `bs:hasTemporalReferenceSystem`
- `bs:hasTemporalReference`
- `bs:TemporalReferenceSystem`(en diverse specialisaties)

- bs :TemporalReference

### 8.1.2 Top Level-attributen

Een attribuut is een kenmerk/eigenschap van een concept en/of individu. De generieke root topAttribuut kent in deze NTA een onderverdeling naar annotaties (door mensen te interpreteren ID's, namen, labels, uitleg, enz.) en door mens en computers te interpreteren kwalitatieve en kwantitatieve aspecten (6.2.8.).

OPMERKING 1 Een essentieel attribuut is een essentiële en oorspronkelijke eigenschap van een concept of individu die niet mag ontbreken, zonder dat het ding ophoudt te zijn wat het is. Dit zou dan een definiërende eigenschap kunnen worden genoemd. Aanvullend bestaan er ook specificerende eigenschappen.

OPMERKING 2 Een attribuut, zoals bijvoorbeeld kleur van een auto of hoogte van een fietsframe, wordt gekenmerkt door een attribuutwaarde. Een attribuutwaarde kan kwalitatief zijn (een 'literal': bijvoorbeeld rood, groen) of kwantitatief (getal met eenheid: bijvoorbeeld 160 cm).

OPMERKING 3 Zowel voor kwalitatieve als kwantitatieve attributen kunnen in een ontologie enumeraties zijn gedefinieerd waarin toegelaten waarden zijn gedefinieerd die gelden binnen een specifieke context.

OPMERKING 4 Een attribuut heeft op instantieniveau een attribuutwaarde die op typeniveau reeds kan zijn voorgedefinieerd ('beperkt').

OPMERKING 5 De ETIM-klasse Dakventilator heeft onder meer de volgende attributen:

- Aansluitspanning, met als attribuutwaarde een keuze uit de enumeratie <1x230V, 3x230V, 3x400V>;
- Luchthoeveelheid bij 50 Pa, met als attribuutwaarde een getalwaarde op schaal ' $\text{m}^3/\text{h}$ ';
- Oppervlaktebescherming waaier, met als attribuutwaarde een keuze uit de enumeratie <Thermisch verzinkt, Sendzimir verzinkt, gecoat>;
- Nom. Kanaaldiameter, met als attribuutwaarde een getalwaarde op schaal 'mm' ;
- Indirecte aandrijving, met als attribuutwaarde 'Ja' of 'Nee' (logisch).

### 8.1.3 Top Level-relaties

#### 8.1.3.1 Inleiding

Als basis voor het bouwen van een ontologie op basis van de top level-concepten worden relaties (in het bijzonder: associaties, zie 6.2.9.) gepresenteerd tussen de top level-concepten onderling. Naast deze associaties is ook hier een root gedefinieerd: de topRelatie.

De associaties verbinden de concepten en zorgen ervoor dat de samenhang van de concepten betekenis en context krijgt. Ze zijn in de context van deze NTA als algemeen toepasbaar erkend en zijn in figuur 20 schematisch weergegeven.

Tot slot worden er drie gespecialiseerde varianten gedefinieerd van het compositie/decompositie-abstractiemechanisme.

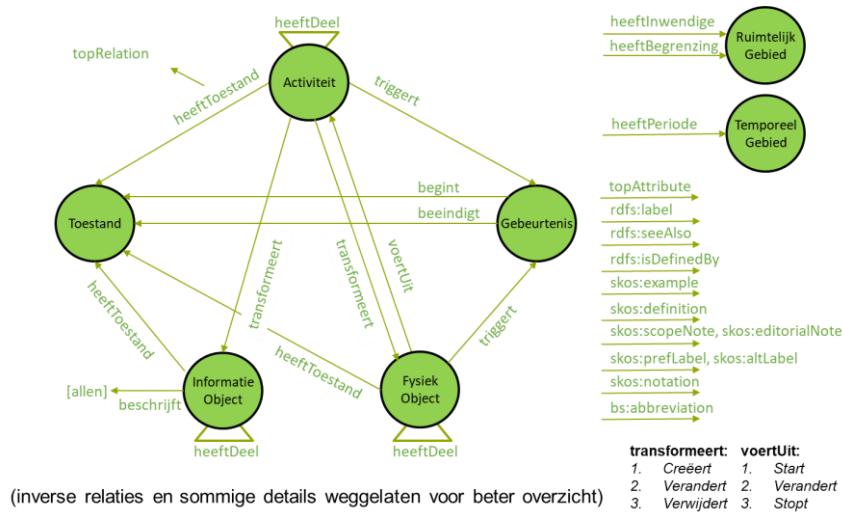
Voor alle associaties en specialisaties van compositie worden ook hun inverse relaties<sup>12</sup> gegeven.

<sup>12</sup> Zoals eerder aangegeven zijn deze inverse relaties niet in de code gemodelleerd (er is hier geen noodzaak toe)

	RELATIE	INVERSE RELATIE
<b>Associaties</b>		
[InformatieObject] beschrijft [FysiekObject]	[Allen] heeftToestand [Toestand]	[Allen] isBeschrevenDoor [FysiekObject]
[FysiekObject] heeftPeriode [Gebeurtenis]	[Gebeurtenis] triggerd [Activiteit]	[FysiekObject] isGetriggeredDoor [Gebeurtenis]
[FysiekObject] voertUit [Activiteit]	[Activiteit] heeftBegrenzing [RuimtelijkGebied]	[FysiekObject] isUitgevoerdDoor [Activiteit]
[FysiekObject] heeftInwendige [RuimtelijkGebied]	[RuimtelijkGebied] heeftBegrenzing [RuinmetelijkGebied]	[FysiekObject] isInwendigeVan [RuinmetelijkGebied]
[FysiekObject] heeftBegrenzing [RuinmetelijkGebied]	[RuinmetelijkGebied] heeftBegrenzing [RuinmetelijkGebied]	[FysiekObject] isBegrenzingVan [RuinmetelijkGebied]
[Activiteit] heeftToestand [Toestand]	[Toestand] heeftBegrenzing [RuinmetelijkGebied]	[Activiteit] isToestandVan [RuinmetelijkGebied]
[Activiteit] transformeert [FysiekObject]	[FysiekObject] heeftBegrenzing [RuinmetelijkGebied]	[Activiteit] isGetransformeerdDoor [FysiekObject]
[Activiteit] transformeert [InformatieObject]	[InformatieObject] heeftBegrenzing [RuinmetelijkGebied]	[Activiteit] isGetransformeerdDoor [InformatieObject]
[Activiteit] triggerd [Gebeurtenis]	[Gebeurtenis] heeftBegrenzing [RuinmetelijkGebied]	[Activiteit] isGetriggeredDoor [Gebeurtenis]
[Activiteit] heeftPeriode [TemporeelGebied]	[TemporeelGebied] heeftBegrenzing [RuinmetelijkGebied]	[Activiteit] isPeriodeVoor [TemporeelGebied]
[Activiteit] heeftInwendige [RuinmetelijkGebied]	[RuinmetelijkGebied] heeftBegrenzing [RuinmetelijkGebied]	[Activiteit] isInwendigeVan [RuinmetelijkGebied]
[Activiteit] heeftBegrenzing [RuinmetelijkGebied]	[RuinmetelijkGebied] heeftBegrenzing [RuinmetelijkGebied]	[Activiteit] isBegrenzingVan [RuinmetelijkGebied]
[Gebeurtenis] begint [Toestand]	[Toestand] begint [Gebeurtenis]	[Gebeurtenis] isBegonnenDoor [Toestand]
[Gebeurtenis] beeindigt [Toestand]	[Toestand] beeindigt [Gebeurtenis]	[Gebeurtenis] isBeeindigdDoor [Toestand]
[State] heeftPeriode [TemporeelGebied]	[TemporeelGebied] heeftPeriode [State]	[State] isPeriodeVoor [TemporeelGebied]
[Toestand] heeftInwendige [RuinmetelijkGebied]	[RuinmetelijkGebied] heeftInwendige [RuinmetelijkGebied]	[Toestand] isInwendigeVan [RuinmetelijkGebied]
[Toestand] heeftBegrenzing [RuinmetelijkGebied]	[RuinmetelijkGebied] heeftBegrenzing [RuinmetelijkGebied]	[Toestand] isBegrenzingVan [RuinmetelijkGebied]
<b>Compositie/decompositie</b>		
[FysiekObject] heeftDeel [FysiekObject]	[FysiekObject] heeftDeel [InformatieObject]	[FysiekObject] isDeelVan [InformatieObject]
[InformatieObject] heeftDeel [Activiteit]	[InformatieObject] heeftDeel [Activiteit]	[InformatieObject] isDeelVan [Activiteit]

Figuur 20 — Top Level-relaties

Wanneer de top level-concepten, attributen, relaties en beperkingen schematisch worden gecombineerd, levert dat figuur 21 op.



Figuur 21 — Top Level-concepten, attributen relaties en beperkingen gecombineerd

In 7.12.3.2 t/m 7.12.3.12 volgt de uitleg en typische voorbeelden voor deze relaties. Tussen haakjes wordt de Engelse term weergegeven.

### 8.1.3.2 beschrijft (describes)

Inverse: isBeschrevenDoor (isDescribedBy)

Deze relatie geldt tussen een Informatie Object en een Fysiek Object, een Activiteit, een Gebeurtenis, een Toestand, een Tijdperiode, een Ruimtelijk Gebied of een ander Informatie Object.

#### VOORBEELDEN

- Neem de Ketelbrug. Daar is in de loop der tijden een waar dossier ontstaan van tekeningen, specificaties, rapporten, nota's, in analoge of in digitale vorm. Elk van deze documenten beschrijft de Ketelbrug en kan worden gezien als een instantie van een Informatie Object die de Ketelbrug beschrijft.
- Een proces-verbaal van een verkeersongeval is een voorbeeld van een Informatie Object dat een éénmalige Gebeurtenis beschrijft.
- De Wikipediapagina over het Twaalfjarig Bestand is een voorbeeld van een Informatie Object dat een individuele Toestand beschrijft.
- Het boek *Dit was het jaar 2018* is een voorbeeld van een Informatie Object dat een individuele Tijdperiode beschrijft.
- Een recensie van Pinkpop 2019 is een voorbeeld van een Informatie Object dat een éénmalige Activiteit beschrijft.
- Een boek over Vlieland is een voorbeeld van een Informatie Object dat een individueel (geografische) RuimtelijkGebied beschrijft.
- Een beschrijving van het kaartblad 52A is een voorbeeld van een Informatie Object dat een individueel Informatie Object beschrijft.

Merk op dat Informatie Objecten alleen individuele instanties kunnen beschrijven van de bovengenoemde Top Level Concepten. Het beschrijven van Fysieke Objecten, Activiteiten, enz. op typeniveau valt er dus niet onder.

### 8.1.3.3 heeftToestand (hasState)

Inverse: isToestandVan (isStateOf)

Deze relatie geldt tussen een Activiteit en een Toestand of tussen een Fysiek Object en een Toestand.

#### VOORBEELDEN

- Neem een pizza. Vóór het bakken bevindt deze zich in de ongebakken toestand, na het bakken bevindt deze zich in de gebakken toestand. Of neem de Ketelbrug. Deze bevond zich vóór het onderhoud in een matige toestand, na het onderhoud in een uitstekende toestand.
- Ook activiteiten kunnen een toestand hebben. De 19<sup>e</sup> etappe van Tour de France 2019 bevond zich na de stillegging ten gevolge van het weer in geneutraliseerde toestand. Het Pleinvrees Festival 2019 bevond zich na de afgelasting in verband met het weer in afgelaste toestand.
- Een toestand kan meerdere parameters bevatten. Zo kan van een stukje atmosfeer zowel de luchtdruk, temperatuur als de luchtvochtigheid worden vastgelegd. Deze drie parameters definiëren dan de toestand van de atmosfeer op een bepaalde locatie en een bepaald tijdstip.

#### 8.1.3.4 triggert (triggers)

Inverse: isGetriggerdDoor (isTriggeredBy)

Deze relatie geldt tussen een Gebeurtenis en een Activiteit en een Gebeurtenis en een Fysiek Object.

##### VOORBEELDEN

- Neem een Verkeersongeval (een gebeurtenis). Deze initieert de komst van de hulpdiensten (een activiteit).
- Een ander voorbeeld is een Meteorietinslag (gebeurtenis). Deze veroorzaakt een Inslagkrater (fysiek object).

#### 8.1.3.5 voertUit (performs)

Inverse: isUitgevoerdDoor (isPerformedBy)

Deze relatie geldt tussen een Fysiek Object en een Activiteit.

##### VOORBEELDEN

- Het Bakken van een pizza (een activiteit) wordt uitgevoerd met behulp van een oven (fysiek object) en door de kok (fysiek object).
- Het Lassen van een brugleuning (een activiteit) wordt uitgevoerd met behulp van een Lasapparaat (fysiek object) en de Lasser (fysiek object).

#### 8.1.3.6 transformeert (transforms)

Inverse: isGetransformeerdDoor (isTransformedBy)

Deze relatie geldt tussen een Activiteit en een Fysiek Object of tussen een Activiteit en een Informatie Object.

##### VOORBEELDEN

- Het Bakken (een activiteit) transformeert een pizza (een fysiek object) van ongebakken (een toestand) naar gebakken (een toestand).
- Het versnipperproces (een activiteit) transformeert een papieren Document (een fysiek object) in papiersnippers (een fysiek object).
- Het doc-to-pdf-conversieproces (een activiteit) transformeert een doc-bestand (een fysiek object) in een pdf-bestand (een fysiek object). Het informatie object verandert niet!

#### 8.1.3.7 heeftPeriode (hasPeriod)

Inverse: isPeriodeVoor (isPeriodFor)

Deze relatie geldt tussen een Toestand en een Tijdperiode of tussen een Fysiek Object en een Tijdperiode.

##### VOORBEELDEN

- Van vrijdag 26 juli tot en met zondag 18 augustus 2019 (tijdperiode) is de N231 (fysiek object) volledig afgesloten (toestand) vanwege groot onderhoud (activiteit).
- Het Twaalfjarig Bestand (toestand) duurde van 9 april 1609 tot 9 april 1621 (tijdperiode).

#### **8.1.3.8 heeftInwendige (hasInterior)**

Inverse: isInwendigeVan (isInteriorOf)

OPMERKING Het doel van de relatie heeftInwendige is om topologische en geometrische relaties tussen Objecten onderling en Activiteiten onderling mogelijk te maken. De notie dat een Object of Activiteit een ruimtelijk gebied "in beslag" neemt vormt het fundament voor deze topologische en geometrische relaties.

Deze relatie geldt tussen een Fysiek Object en een (topologisch) Ruimtelijk Gebied, tussen een Activiteit en een Ruimtelijk Gebied en tussen een Toestand en een Ruimtelijk Gebied.

Het gaat hier over de relatie tussen het FysiekObject, Activiteit, Toestand en een RuimtelijkGebied als geheel, dus beide als 3D-objecten/volumes. Het gaat dus nog niet over een locatieriferentie (zoals bij een coördinaat), waarbij de locatie van een specifiek 0D-/1D-/2D-/3D-geometrisch aspect van het FysiekObject wordt gespecificeerd ten opzichte van referentiepunten van het bijbehorende locatieriferentiesysteem. Als gevolg hiervan is er bij heeftInwendige geen aangrijppingspunt, anders dan het gehele 3D-volume.

#### VOORBEELDEN

- De Boekenkast (fysiek object) staat in de rechterachterhoek van de woonkamer (ruimtelijk gebied).
- Pinkpop (activiteit) vindt plaats op Megaland Landgraaf (ruimtelijk gebied).
- Het Twaalfjarig Bestand (toestand) gold voor het gebied van de Zeven Verenigde Provinciën (ruimtelijk gebied).

Merk op dat een bepaald ruimtelijk gebied niet wordt gedecomponeerd (met behulp van hasPart) in andere ruimtelijke gebieden, maar dat een ruimtelijk gebied alleen topologische relaties kan hebben met andere ruimtelijke gebieden. Zo omvat een wegvak (een fysiek ruimtelijk gebied) een aantal wegstroken (fysieke ruimtelijke gebieden). Deze wegstroken grenzen aan elkaar.

Merk op dat ruimtelijke gebieden in veel gevallen geen eigen identificatie hebben. Vaak wordt het fysiek object waarmee het gebied is verbonden, gebruikt als identificatie van dat gebied. Zo kan de N231 worden beschouwd als een FysiekObject (verharding, kunstwerken, enz.), maar ook als een Verkeersruimte waarbinnen voertuigen zich kunnen voortbewegen.

Merk verder op dat voor ruimtelijke gebieden die voor hun representatie gebruikmaken van een coördinatenstelsel, het belangrijk is om aan te geven welk referentiesysteem er is gebruikt. In 8.1.1.7 is een voorbeeld gegeven.

#### **8.1.3.9 heeftBegrenzing (hasBoundary)**

Inverse: isBegrenzingVan (isBoundaryOf)

Deze relatie geldt tussen een Fysiek Object en een Ruimtelijk Gebied.

## NTA 8035:2020

**OPMERKING** Het doel van de relatie heeftBegrenzing is om topologische en geometrische relaties tussen Objecten onderling en Activiteiten onderling mogelijk te maken. De notie dat een Object of Activiteit door een ruimtelijk gebied wordt begrensd, vormt het fundament voor deze topologische en geometrische relaties.

### VOORBEELDEN

- Een beukenhaag (fysiek object) vormt de grens van het perceel van T. Janssen (ruimtelijk gebied) en het perceel van G. Pieterse (ruimtelijk gebied).
- De woonkamer (ruimtelijk gebied) wordt begrensd door vier muren (fysieke objecten), de vloer (fysiek object) en het plafond (fysiek object).

Ruimtelijke gebieden die worden begrensd door Fysieke Objecten, worden Fysieke Ruimtes genoemd. Een Fysieke Ruimte is daarmee een subtype van Ruimtelijk Gebied, dat ook subtypen kan bevatten als Administratief Gebied, Functioneel Gebied, enz.

### 8.1.3.10 begint (begins)

Inverse: isBegonnenDoor (isBeginBy)

Deze relatie geldt tussen een Gebeurtenis en een Toestand.

### VOORBEELDEN

- Met de ondertekening (gebeurtenis) begint het Twaalfjarig Bestand (toestand).
- De blikseminslag (gebeurtenis) is het begin van de bosbrand (toestand).
- Met het doorknippen van het lint werd de nieuwe weg geopend (toestand).

### 8.1.3.11 beëindigt (ends)

Inverse: isBeeindigdDoor (isEndedBy)

Deze relatie geldt tussen een Gebeurtenis en een Toestand.

### VOORBEELDEN

- De inval van Polen (gebeurtenis) beëindigde de vrede (toestand).
- Het onweer (gebeurtenis) beëindigde het mooie weer (toestand).

### 8.1.3.12 heeftDeel (hasPart) beperkingen

Inverse: isDeelVan (isPartOf)

Deze relatie geldt tussen Fysiek Objecten onderling, tussen Informatie Objecten onderling of tussen Activiteiten onderling.

### VOORBEELDEN

- De fiets van Ellen (fysiek object) bestaat uit een frame, twee wielen en diverse andere onderdelen (alle fysieke objecten).

- Hoofdstuk 4 (informatie object) bestaat uit 4.1 en 4.2 (informatie objecten).
- Het bereiden van een pizza (activiteit) bestaat uit het snijden van de ingrediënten, het maken van de pizzabodem, het configureren van de pizza, en het bakken van de pizza (alle activiteiten).

#### **8.1.4 Implementatie van het Top Model**

De implementatie van het Conceptuele Top Model is toegevoegd aan de implementatie van het Conceptueel Meta Model in OWL (zie bijlage D).

- 1) De Top Level-concepten worden OWL-klassen (behalve TopConcept).
- 2) De Top Level-attributen worden OWL data type properties (behalve topAttribuut).
- 3) De Top Level-relaties worden OWL object properties (behalve topRelatie).
- 4) De Top Level-annotaties waren al aanwezig via de SKOS/RDFS imports.

De root items mappen (optioneel want ook afleidbaar) bij OWL op de volgende wijze naar de door OWL voorgedefinieerde taalconstructies:

- owl:Thing als TopConcept;
- owl:topDataProperty als topAttribuut;
- owl:topObjectProperty als topRelatie.

OPMERKING 1 Naast de OWL Top Level-klassen van zijn er nog meer modelleer-technische OWL-klassen als gevolg van de objectificatie (bsc:Property) en als gevolg van de representatie van enumeratielijsttypen (bs:EnumerationType), grootheden en eenheden.

OPMERKING 2 In een eindgebruikersontologie worden al deze klassen en (data en object) properties verder gespecialiseerd, gedecomposeerd, uitgebreid, geannoteerd, beperkt, enz. De extra conceptklassen voor eindgebruikers zullen steeds specialisatie zijn van de bestaande zeven voorgedefinieerde conceptklassen, omdat deze als volledig dekkend worden gezien.

#### **8.2 Complexe attributen/relaties ('properties' in RDF)**

De default-modellering van attributen en relaties in deze NTA kan het eenvoudigst plaatsvinden door direct gebruik te maken van de relevante taalconstructies gegeven door RDF, RDFS, OWL en SHACL: data type properties voor attributen en object properties voor relaties.

Soms zijn echter meer expliciete metagegevens nodig, niet alleen voor kwantitatieve attributen, maar ook voor kwalitatieve attributen en voor relaties. Een andere wens is de mogelijkheid om eigenschappen te modelleren die bestaan uit andere eigenschappen.

Voor dergelijke 'complex' attributen en relaties wordt een aanpak gebruikt die is gebaseerd op de Ontology for Property Management (OPM) ([22]). OPM definieert drie niveaus. Niveau komt overeen met onze simpele modeleerwijze. Niveau 2 voegt één indirecte toe via objectivering van toegewezen attributen/relaties met een waarde en niveau 3 voegt een tweede objectivering toe voor de toekenning van de eigenschapswaarde.

Onze complexe attribuut/relatie-benadering is gebaseerd op OPM niveau 2. Wel is de benadering aangepast:

## NTA 8035:2020

- OPM gebruikt CDT/UCUM voor grootheden en eenheden. In deze NTA wordt consistent QUDT (versie 2.1) gebruikt;
- De complexe variant in deze NTA hergebruikt de simpele variant voor de definitie van de properties (OPM specialiseert hier toe bsc:Property);
- We maken meer gebruik van bestaande taalconstructies (rdf:value, rdfs:Container, rdfs:member).

Alle detailinformatie hierover is opgenomen in bijlage G.

**OPMERKING** Een voorbeeldbestand voor complexe **kwantitatieve** attributen is opgenomen in bijlage H. Een voorbeeld van een complexe relatie is opgenomen in bijlage I. Een voorbeeld van een complexe relatie ten behoeve van enumeratietypen is opgenomen in bijlage J.

## 9 Toepassing bij gegevensintegratie

### 9.1 Integratie scenario's

Het traditionele integratiescenario gaat uit van gegevensuitwisseling. Partij A heeft data die partij B wil hebben, en partij A stuurt deze file- of container-gebaseerd naar partij B. Vaak is het hierbij onduidelijk of er ook eigenaarschap (bronhouderschap) overgaat. In elk geval zijn er opeens twee kopieën van de data. Dit kan onduidelijkheid geven over wat de waarheid is, zeker bij veranderingen op de gegevens in een later stadium.

Daarom is er steeds vaker behoefte aan een Single Source of Truth (SSoT), ook wel aangeduid met het motto 'eenmalige vastlegging, meervoudig gebruik'. Dit kan worden gerealiseerd door data niet meer uit te wisselen maar bij de bron te laten staan en benaderbaar te maken, oftewel datadeling.

Deze NTA kan voor beide scenario's worden gebruikt, zoals in dit hoofdstuk verder is uitgewerkt.

### 9.2 Gegevensuitwisseling

Er kunnen volgens ISO 15926-11 vijf abstractielagen worden onderscheiden bij gegevensuitwisseling (zie figuur 22):

- De bovenste laag (laag 1) geeft de rol weer die een specifieke organisatie of onderneming speelt bij een transactie/interactie, zoals het uitwisselen van informatie, en wordt de oorsprong en reikwijdte van de informatie gedefinieerd.
- De inhoudlaag (laag 2) vertegenwoordigt de betekenis van de objecten die worden uitgewisseld zoals gedefinieerd door een overeengekomen ontologie.
- De semantische laag (laag 3) wordt gedefinieerd door het in deze NTA gedefinieerde Conceptuele Meta Model gebonden aan een taal, zoals OWL.
- De syntaxlaag (laag 4) vertegenwoordigt gegevensformaten. Deze NTA gaat uit van het gebruik van W3C RDF-serialisatiemethodes maar schrijft geen specifieke methode voor.
- De transportlaag (laag 5) beschrijft de technologie die wordt gebruikt om de gegevens geschikt te maken voor overdracht (files, containers).

Binnen een specifiek project kan elke laag nader worden gespecificeerd in een projectspecifieke informatieleveringsspecificatie (ILS). In zo'n handleiding voor informatielevering kunnen eisen,

**Met opmerkingen [wb13]:** Bedoelen jullie:

NEN-ISO/TS 15926-11, Industrial automation systems and integration - Integration of life-cycle data for process plants including oil and gas production facilities - Part 11: Methodology for simplified industrial usage of reference data

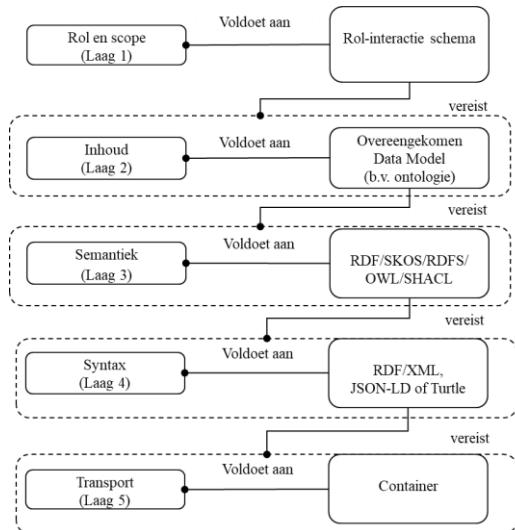
?

Ik kan ISO 15926-11 verder niet vinden bij NEN.  
Graag opnemen in h 2. (En schrappen in bibliografie.)

**Met opmerkingen [BH(14R13):** moet gevraagd aan leo

**Met opmerkingen [BH(15R13):** idem. zie vorige antwoord...dus JA, die bedoelen we...graag zo verwerken

beperkingen, specifieke keuzes of uitbreidingen worden opgenomen met betrekking tot de functionaliteit en implementatie van elke laag.



**Figuur 22 — Lagenmodel voor gegevensuitwisseling tussen partijen**

Het proces van het uitwisselen van projectgegevens kan worden opgesplitst in verschillende deelprocessen (zie figuur 23). Het proces aan de linkerzijde is een eerste vereiste bij het starten van het uitwisselen van gegevens met andere partijen: hiertoe moet als eerste een keuze worden gemaakt met betrekking tot de te hanteren ontologie.

Van links naar rechts zorgt het eerste proces ervoor dat alle relevante gegevens in de files of databases van de verzendende partij correct worden gestructureerd volgens de gezamenlijk overeengekomen ontologie.

In het tweede proces wordt een selectie gemaakt van welke gegevens (een reeks individuele dingen, bijvoorbeeld met betrekking tot een specifiek onderwerp en volgens een specifieke baseline in een project) moeten worden voorbereid om te 'overhandigen' aan de andere partij.

Vervolgens begint het proces waarin, afhankelijk van het niveau van 'capabilities' (zie paragraaf 7.2), de geselecteerde gegevensverzameling semantisch wordt gecodeerd volgens de RDFS- respectievelijk OWL/SHACL-wijze zoals eerder beschreven paragraaf 7.3 t.m. 7.6.

**Met opmerkingen [BH(16): idem.**

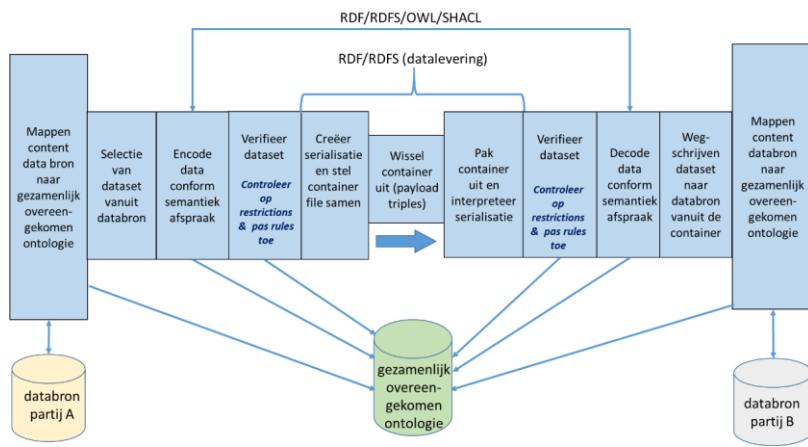
Deze dataset moet worden geverifieerd aan de hand van de beperkingen en regels die enerzijds zijn gedefinieerd in de gemeenschappelijk overeengekomen ontologie en anderzijds zijn gedefinieerd in de handleiding voor informatieaflevering, opgesteld door de vragende partij.

Vervolgens start het proces van serialisatie op basis van een overeengekomen serialisatiemethode (bijvoorbeeld Turtle of RDF XML Syntax) en worden de relaties eventueel voorzien van relevante metagegevens. Dit proces eindigt met een bestand dat alle payload triples vertegenwoordigt.

Wanneer de andere partij de container voor gegevensuitwisseling ontvangt, worden deze processen in tegenovergestelde richting uitgevoerd:

## NTA 8035:2020

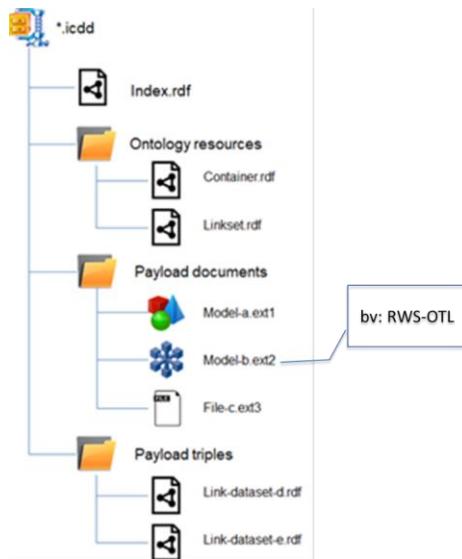
- het uitpakken van de container met payload data/documenten;
- verificatie van de gegevens aan de hand van de beperkingen en regels zoals gedefinieerd in de ontologie en de handleiding voor informatielevering;
- decoding naar de gebruikte serialisatiemethode;
- opslag van de overgedragen gegevens in de files/databases van de ontvangende partij volgens de mapping tussen het 'native' gegevensmodel en de gedeelde ontologie.



**Figuur 23 — Processtappen bij data-uitwisseling tussen twee partijen (van links naar rechts)**

Een ontologie functioneert hierbij dus als een gemeenschappelijk gedeelde semantische referentie voor alle partijen die samenwerken in een project en projectgegevens willen uitwisselen/delen op een neutrale en ondubbelzinnige manier.

De container als pakketmechanisme voor transport moet voldoen aan ISO 21597-1 (ICDD). In figuur 24 is de structuur van deze ZIP file weergegeven.



**Figuur 24 — Containerstructuur**

Met een container kunnen zowel gestructureerde gegevens en gegevensmodellen, zoals ontologieën als de RWS-OTL, als niet-gestructureerde documenten worden uitgewisseld. In het algemeen is er een hybride situatie van beide.

ICDD gebruikt LD/SW om metadata vast te leggen over een container, zijn payload met documenten en hun onderlinge (generieke, geobjectificeerde) relaties, inclusief deep links tussen elementen in deze documenten. In de toekomst zal ISO 21597-2 het mogelijk maken om de documenten en hun generieke links uit ISO 21597-1 nadere semantiek mee te geven.

Het blijft hierbij echter gaan over relaties tussen gegevensverzamelingen, gegevensmodellen, enz., en eventueel elementen uit deze verschillende documenten. De relaties binnen deze documenten worden bepaald door deze NTA.

In de ontwikkeling van ISO 21597-2 wordt zoveel mogelijk aangestuurd op afstemming tussen de relevante relaties (simpelere varianten in NTA, complexe/geobjectificeerde varianten in ISO 21597-2).

### 9.3 Gegevensdeling

Bij gegevensdeling worden er geen gegevens uitgewisseld maar gepubliceerd en wel op zo'n manier dat ze in principe (afhankelijk uiteraard van authenticatie/autorisatie) door andere partijen benaderbaar zijn.

Dus in plaats dat er een (container)file wordt gestuurd van partij A aan partij B, zet partij A deze op een (web)plek waar partij B erbij kan. Kortom, eenmalige vastlegging, meervoudig gebruik.

Deze publicatie dient net als bij de file-uitwisseling in een RDF-formaat en volgens een afgesproken ontologie te gebeuren. Eventueel kan de container nog steeds als pakketmechanisme worden gehanteerd.

#### 9.4 Linken tussen gegevens en tussen gegevensstructuren

Het modelleren van ontologieën en bijbehorende dataverzamelingen is één ding, het aan elkaar relateren/verbinden door linken van zowel ontologieën als datasets is een ander. In de toekomst wordt dit linken steeds belangrijker omdat alle ontologieën en datasets met elkaar verband houden en elkaar kunnen aanvullen. Het idee is dat er netwerken of ecosystemen van data en datastructuren ontstaan die als digitaal stelsel de basis gaan vormen voor sectorbrede digitalisatie, integratie en innovatie.

Dit linken kan op diverse manieren plaatsvinden:

- impliciet via bijvoorbeeld een SPARQL query die meerdere ontologieën en/of datasets vraagt. De links zitten dan als het ware impliciet gecodeerd in de SPARQL-code.
- expliciet (en daardoor beter herbruikbaar) door declaratief de links te modelleren, liefst apart van de te linken ontologieën/datasets in linksets. Op deze manier blijven de linksets/items zelf ‘agnostic’ en ‘clean’.

Deze linksets zijn verzamelingen relaties en zijn er in twee soorten: simpel en complex (geobjectificeerd).

Technisch gezien, vanuit een linked data-technologieperspectief, is een simpele linkset een verzameling RDF triples waarbij het object en de subject van de triples refereren naar elementen uit verschillende RDF-datasets (die of een ontologie of een dataset representeren). lkef:Property kan daarbij als link worden gezien; dus eigenlijk alle attributen/relaties die bij het modelleren worden gebruikt:

- 1) disk:Brug rdfs:subClassOf cbnl:Overspanning ;
- 2) bdb:VanBrienenoord owl:sameAs cdb:Brienenoord ;
- 3) ex:GebouwTNO-SW bs:hasPart bim:Room-E3.64 ;
- 4) rot:MaasTunnel rdf:type rwsot:Tunnel ;
- 5) ex1:Asfalt ex2:getestDoor kvk:TNO .

Het wordt echter aanbevolen om waar mogelijk relaties te gebruiken die op taalniveau (RDF, SKOS, RDFS, OWL, SHACL) semantisch volledig formeel zijn gedefinieerd en daardoor ook voor bijvoorbeeld ‘reasoning’ kunnen worden gebruikt. In de opsomming zijn dit voorbeelden 1), 2) en 4).

Ook de voorgedefinieerde Top Level-relaties kunnen met voors en tegens voor linking worden gebruikt.

#### Voordelen

- De links zijn concreter.
- Decompositie komt beschikbaar.
- De complexe varianten kunnen metadata omvatten (Wie heeft de link aangebracht? Met welk doel? enz.).

#### Nadeel

- De linksets worden afhankelijk van deze NTA. Er zijn veel situaties waarbij deze NTA of zijn Europese CEN SMLS-tegenhanger ([5]) nog niet in beeld is. Dan is een generieke taalniveau-oplossing beter.

### Linksets op taalniveau

De mogelijkheden voor koppeling van ontologieën (onderling) en hun bijbehorende gegevensverzamelingen (onderling) hangt in hoge mate af van het gehanteerde modeleerniveau. De volgende taalconstructies voor elk niveau kunnen handig worden gebruikt:

LoC-1 Weak linking:

- skos:exactMatch;
- skos:closeMatch;
- skos:narrowMatch;
- skos:broadMatch;
- skos:relatedMatch;
- rdfs:seeAlso.

LoC-2 Medium linking:

- rdfs:subClassOf tussen klassen;
- rdfs:subPropertyOf tussen properties;
- via rdfs:domain of rdfs:range (dat wil zeggen: verwijzing naar een domein of range-klasse/datatype in een externe ontologie).

LoC-3 Strong linking:

- LoC-2-linken hergebruikt voor klasse-niveau:
  - + de shorthands: owl:equivalentClass en owl:equivalentProperty.
  - owl:sameAs tussen individuen op gegevensniveau.

Koppelingen op dataniveau worden bij voorkeur vermeden door één individu met meerdere rdf:type-relaties naar RDFS/OWL-klassen uit verschillende ontologieën te classificeren (eenmalige vastlegging). Merk op dat dit niet altijd mogelijk is, omdat er vaak sprake is van meerdere onafhankelijke gegevensbronnen/bronhouders die iets zeggen over dezelfde verschijnselen in de echte wereld.

Het is moeilijk om metadata aan de triples in een simpele linkset te koppelen. Als hier behoefte aan is, dan moeten de links worden geobjectificeerd.

### Link met ISO ICDD

Ook ISO 21597-1 (deel 1 van ISO ICDD), dat naast een pakket (container met een payload) van documenten ook geobjectificeerde links tussen elementen in deze documenten specificert (de zogenoemde 'deep links'), biedt een geobjectificeerd Link-concept (ls:Link). ISO 21597-1 zal in 2020

**Met opmerkingen [BH(17): check, ok zo]**

## NTA 8035:2020

worden gepubliceerd als International Standard (IS). Bij het schrijven van deze NTA is de status nog 'Final Draft International Standard (FDIS)'.

ICDD richt zich in principe (volgens het toepassingsbereik, de scope) alleen op data-uitwisseling (Engels: delivery) niet op datadeling. Zowel de containerbeschrijving en de document- en linkbeschrijvingen zouden echter ook prima in de context van datadeling kunnen worden hergebruikt.

Ook richt de ICDD zich niet specifiek op LD/SW-bestanden IN de payload (wel voor het beschrijven VAN de payload), maar meer op een hybride situatie waarin ongestructureerde documenten en gestructureerde documenten (waaronder eventueel RDF-datasets en ontologieën) aan elkaar worden gekoppeld.

In de toekomst komt er ook een ISO 21597-2, dat meer specialisaties voor de links uit deel 1 gaat bieden. Als die er te zijner tijd zijn, en als deze voldoende in lijn zijn met de behoeften en huidige manier van modelleren van complexe relaties, dan worden ze opgenomen in een nieuwe editie van deze NTA.

### 9.5 Versiebeheer

Deze NTA bevat geen specifieke aanbevelingen met betrekking tot versiebeheer. Wel bevat het enkele aandachtspunten en mogelijkheden. Versiebeheer is mogelijk op verschillende granulariteitsniveaus:

- 1) versie van complete gegevensverzamelingen;
- 2) versie van een deel van gegevensverzameling, bestaande uit individuen;
- 3) versie van een volledig gegevensmodel als een ontologie;
- 4) versie van een deel van een gegevensmodel;
- 5) versiebeheer op triple-niveau als het kleinste (atomaire) gegeven (statement) binnen een gegevensmodel of gegevensverzameling.

Voor niveaus 1), 2), 3) en 4) kan elk gepubliceerde gegevensmodel of elke gepubliceerde gegevensverzameling of elk fragment/element hiervan een versienummer krijgen volgens een expliciet coderingssysteem.

Het gebruik van versies binnen URI's wordt in het algemeen afgeraden. URI's moeten vooral stabiel zijn en daartoe zo weinig mogelijk semantiek bevatten. Voor niveau 4) is in [9.7.1] wel een voorbeeld gegeven. Stabiliteit wordt hierbij gcompenseerd door een 'laatste versie'-concept.

Versiebeheer volgens niveau 5) houdt in dat versiebeheer op een of andere wijze op triple-niveau kan worden gerealiseerd. In bijlage B (RDF) worden diverse mogelijkheden gegeven om triple level-metadata zoals versies toe te voegen door middel van reïficatie, singuliere properties, named graphs-principegebruik, objectificatie, enz.

Binnen een specifiek project kan de gewenste versiestrategie worden geadresseerd als onderdeel van een projectspecifieke informatieleveringsspecificatie (ILS).

**Met opmerkingen [wb18]:** Graag correcte verwijzing.

**Met opmerkingen [TH19R18]:** Tom loopt na

## 10 Conformiteit

### 10.1 Conformiteit in het algemeen

De gegevens zijn geformatteerd in een officiële RDF-serialisatie, zoals bijvoorbeeld:

- RDF 1.1 XML Syntax;
- RDF 1.1 Turtle;
- JSON-LD.

In deze NTA wordt het meest gebruikgemaakt van Turtle, de meest mensvriendelijke variant. Bijlage C bevat een korte introductie van Turtle.

De gegevens zijn gestructureerd volgens een gegevensmodel opgesteld volgens een van de drie beschreven modelleerniveau's, waarbij wordt gebruikgemaakt van de taalconstructies zoals gegeven door de bijbehorende modelleerstijl(en)/taalbindingen (zie hoofdstuk 7). Voor niveau 3 geldt de extra keuze tussen OWL en SHACL ('open versus closed world'-aanname).

Dit gegevensmodel maakt waar mogelijk gebruik van het Conceptueel Top Level Model (volgens dezelfde taalbinding/modelleerstijl(en)) (zie hoofdstuk 7).

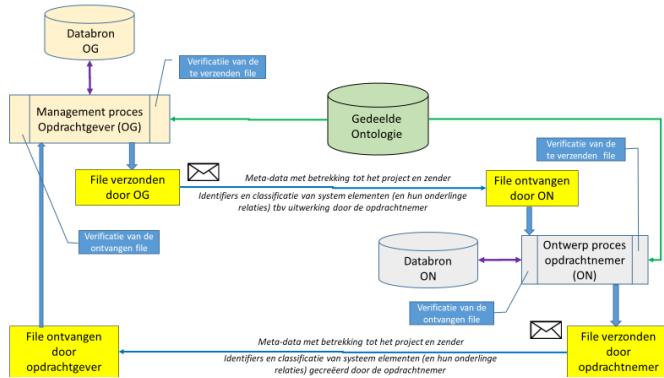
Het gegevensmodel maakt waar mogelijk gebruik van de modelleerpatronen (zie hoofdstuk 8). Een speciale keuze is hier het complexe attribuut/relatie-patroon als alternatief voor het default simpele patroon.

### 10.2 Specifiek bij (file-gebaseerde) uitwisseling van gegevens

In het geval er een pakket van (onderling gerelateerde) documenten (inclusief gegevensverzamelingen en eventueel gegevensmodellen) wordt uitgewisseld, dan wordt de ISO ICDD-container (ISO 21597-1 en ISO 21597-2<sup>13</sup>) toegepast. Dit betekent onder andere dat de metagegevens van de documenten en de interrelaties (inclusief deep links tussen elementen in de documenten) worden vastgelegd middels standaard container- respectievelijk linksetontologieën. De momenten van gegevensverificatie in het proces van gegevensuitwisseling zijn geschetst in figuur 25.

---

<sup>13</sup> Deze standaarden zijn in voorbereiding



**Figuur 25 — Voorbeeld uitwisseling van ontwerpgegevens opdrachtgever en opdrachtnemer**

### 10.3 Specifiek bij (web-gebaseerde) deling van gegevens

Bij de benadering van gegevens kan worden gebruikgemaakt van de standaard (RDF-level) W3C SPARQL-vraagtaal. Een file- of containerformaat is in dat geval vaak niet relevant meer.<sup>14</sup> Wel moet er worden gezorgd dat de URI's die zijn gebruikt in de gegevens of gegevensstructuur dereferenceable zijn. Dit betekent dat de gegevens ook via die identifiers kunnen worden gelokaliseerd.

Naast SPARQL zijn er diverse andere opties om te 'interfacen' beschikbaar/in ontwikkeling. Vaak hebben ze het doel om het leven van ontwikkelaars van semantische software makkelijker te maken door SPARQL-complexiteit te verbergen.

#### VOORBEELDEN

- <https://open-services.net/>;
- <http://grlc.io/>, generatie van RESTful API's op basis van SPARQL queries ;
- OGC/W3C WFS3, [https://github.com/opengeospatial/WFS\\_FES](https://github.com/opengeospatial/WFS_FES), <http://docs.opengeospatial.org/DRAFTS/17-069r3.html> ;
- Trifid, <https://zazuko.com/products/trifid/>, HTML view in JavaScript gebaseerd op JSON-L output van SPARQL queries ;
- W3C Linked Data Templates (LDT), <https://atomgraph.github.io/Linked-Data-Templates/>, <https://www.w3.org/community/declarative-apps/> ;
- GraphQL, <https://graphql.org/>, een vraagtaal naast 'beyond' SPARQL die makkelijker te leren en toe te passen is. Er zijn specifieke implementaties die op linked data werken (bijvoorbeeld in TopQuadrant's TopBraid Composer) ;
- Voor de toekomst: SQL-based ISO Graph Query Language (GQL): <https://www.gqlstandards.org/> .

Deze interfaces kunnen worden gebruikt voor het benaderen van op deze NTA-gebaseerde data en datastructuren. Een goed voorbeeld van deze toepassing is de website [apps.gsw.nl](http://apps.gsw.nl), een verzameling

<sup>14</sup> Hoewel het nog steeds mogelijk is om het pakket (container) ook toe te passen bij publicatie.

GWSW-applicaties die voorziet in toegang tot de GWSW-server. Ze verzorgen de upload, toetsing en presentatie van (gemeentelijke) datasets.

#### **10.4 Informatieleveringsspecificatie (ILS)**

Een ILS (bijvoorbeeld bij een contract) kan aangeven welke exacte conformiteit voor de data-uitwisseling of -deling is geëist/gewenst omtrent:

- gegevenstransportmechanisme: uitwisseling of deling;
- wel of geen packaging (denk ICDD container en/of VISI envelop);
- syntax form van de gestructureerde gegevensverzamelingen (bijvoorbeeld RDF/XML, Turtle of JSON-LD);
- gehanteerde logische interpretatiewijze: ‘open world’- of ‘closed world’-aanname;
- ambitieniveau (taalbinding): SKOS, RDFS, OWL of SHACL of combinatie (en ...);
- simpele of complexe attributen, of beide;
- gebruik van standaard Top Level-concepten/-annotaties/-relaties/-restricties;
- verdere details/specialisaties van huidige Top Level;
- soort versiebeheer.

Wellicht nader gespecificeerd aan de hand van Conformance Classes (CC's).

### **11 Bekende issues**

De volgende issues zullen aan de orde komen in een update van deze NTA (NTA 2.0):

- Directe referentie zonder import uit de lokale workspace of on-the-fly import vanuit het web van QUDT-ontologieën (qudt.ttl, quantitykind.ttl, unit.ttl en hun imports) gaat nog niet goed in TopBraid Composer versie 6.3.0. Na het downloaden van alle benodigde ontologieën en opslag in lokale workspace gaat het wel goed. De QUDT ontologieën worden nu meegeleverd in de code bij deze NTA.
- Nadere analyse van de huidige taalbinding LoC-1 (SKOS) met betrekking tot de attributen en relaties. Is een mapping naar rdf:Property in plaats van skos:Concept relevant (nu naar skos-concepten)? Welke rol speelt skos:notation hierbij?
- Er is nu geen mogelijkheid voor decompositie van attributen. Complex attribuut instanties kunnen wel decomponeren maar een typische (definitieniveau) decompositie via restricties aangeven is niet mogelijk voor datatype properties.
- Er zijn nu een code files voor LoC-3a (OWL): basicsemantics-owl.ttl en basicsemantics-owl-complex (zie bijlage D). In NTA 2.0 hebben alle andere LoCs ook een variant:
  - basicsemantics-skos.ttl, basicsemantics-skos-complex.ttl ;
  - basicsemantics-rdfs.ttl, basicsemantics-rdfs-complex.ttl ;

— basicsemantics-shacl.ttl, basicsemantics-shacl-complex.ttl .

— Mogelijke technologie-verbeteringen:

— Er is een grote kans dat RDF opgevolgd gaan worden door RDF\* (“RDF star”) ([24]) met bijbehorende Turtle\* voor serialisatie en SPARQL\* voor de directe benadering. Met RDF\* wordt het mogelijk om een triple als geheel weer als object van een andere triple op te nemen. Oftewel, geneste triples worden mogelijk. Op deze wijze kan meta-data over de triple (en dus het predicaat van de triple) op een standaard wijze gemodelleerd worden. Dan is objectificatie van het predicaat niet meer nodig en wordt het complex property patroon overbodig. Op eenzelfde wijze wordt het dan waarschijnlijk mogelijk om meta-data over de predicaatdefinitie op te nemen door bijvoorbeeld de triple “ex:height rdf:type owl:DatatypeProperty” weer als object op te nemen. Hiermee voorkomen we wellicht OWL Full. Dit behoeft nader onderzoek en uitwerking. Semantische tools als TopBraid Composer en Stardog hebben reeds experimentele RDF\* ondersteuning.

Met opmerkingen [BH(20): nieuw

— Mogelijke CMM-verbeteringen:

— meer taxonomische structuur, bijvoorbeeld een MetaConcept klasse als ‘root’;

— Attribuut/Relatie-instanties toevoegen (afsplitsen)? Niet relevant voor RDF-gebaseerde taalbindingen;

— Checken of het koppelen van beperkingen en afleidingen aan concepten, attributen en relaties ook een vorm van karakterisering is.

— Mogelijke CM-verbeteringen:

— toevoegen topologische relaties bij concept RuimtelijkGebied? (Bijvoorbeeld de relatie bs:isConnectedTo);

— Betere termen voor heeftInwendige en heeftBegrenzing (minder topologisch, het gaat om de link van een fysiek object of activiteit naar een topologische/geometrische ruimte)?;

— misschien twee SE-subklassen toevoegen voor PhysicalObject: FunctionalPhysicalObject en MaterializedPhysicalObject en hun onderlinge relatie (Engels: fulfilment of realizes) om transitieve/geneste functie/materiaal-decompositie mogelijk te maken;

— misschien heeftInvoer/heeftUitvoer-relaties toevoegen bij Activiteit;

— aangeven hoe om te gaan met volledig gedefinieerde Types op plaatsing in ruimtetijd na (via subklassen of via gerelateerde instanties). Vergelijk bSI IFC-aanpak met “XType” entiteiten in EXPRESS;

— de Top Level-relatie ‘beschrijft’, enz., is voor instanties van Top Level-concepten; maar wat moet er gebeuren met beschrijvingen voor de concepten zelf? (of is dat al voldoende geregeld via de geselecteerde annotaties?);

— met betrekking tot modelleerpatroon Complex Properties:

— algemeen: eventuele update van het modelleerpatroon voor complexe attributen en relaties als gevolg van afstemming met W3C LBD, bSI en/of CEN TC442 WG4 Product Data Templates (PDT's);

## NTA 8035:2020

- de term bsc:hasProperty wordt door sommigen als verwarrend. De term hasProperty lijkt namelijk alleen over eigenschappen te gaan. Omdat property hier de RDF-betekenis heeft zijn ook relaties gedekt en klopt het wel. Misschien is het beter om bsc:hasAspect te gebruiken. Maar dan ook bsc:Aspect in plaats van bsc:Property, enz. Nu is een 'aspect' een generalisatie van kwantiteit en kwaliteit; dat moet dan ook anders.
- Versiemanagement (op zowel data- als ontologieniveau);
  - er zijn nog te veel mogelijke benaderingen: impliciet (functioneel) of expliciet (in de data), op verschillende detailniveaus (container, model/document, element, triple) of op metadata. Verder is er een verschil bij beheer en publicatie/gebruik, de rol van de URI-strategie hierbij, enz. Er is hierover nog geen consensus. Iedereen hoopt eigenlijk op een standaard van boven (W3C?). Tegelijkertijd heeft wel iedereen met de problematiek te maken en zou men graag een standaard of 'best practices' hiervoor volgen. Hopelijk kan hierover in een volgende editie van deze NTA meer over afspreken;
  - er lijkt grote behoefte te bestaan aan een neutrale, expliciete modellering, niet alleen een onderliggende, specifieke implementatieoplossing. Interessante optie is de huidige CROW-aanpak bij IMBOR-OTL: de gemodelleerde complex property-instanties krijgen twee metadata-attributen: validFrom en validThrough. GWSW hanteert een soortgelijke aanpak, maar dan op basis van de laatste observatie (eigenlijk alleen validFrom).
- Mogelijke verbreding van de NTA
  - deze NTA is o.i. breed toepasbaar in de context van de gehele gebouwde omgeving sector. Ook is deze onafhankelijk van bestaande concepten als BIM en GIS. Wel is er een nadruk geweest in de ontwikkeling vanuit de Infra en BIM hoek. We zijn dan ook in gesprek met partijen die betrokken zijn bij soortgelijke activiteiten vanuit de GIS wereld (Geonovum, Kadaster, Vereniging van Nederlandse Gemeenten (VNG) met hun Meta Informatie Model (MIM) initiatief dat in de laatste versie ook nadrukkelijk Linked Data betreft. Aan deze GIS zijde speelt UML vaak een prominente rol (vanuit historie maak ook vanuit de praktische link met bestaande applicatielandsc
  - happen). Ook de B&U sector zal nadrukkelijker betrokken worden bij de verdere ontwikkeling van deze NTA.

Met opmerkingen [BH(21): nieuw

**Bijlage A**  
(normatief)

**W3C-taal deelverzamelingen**

De exacte deelverzamelingen van de bestaande W3C Linked Data/Semantic Web-talen die in deze NTA worden gehanteerd, worden in deze bijlage gedefinieerd.

**XML Schema Part 2: Datatypes**

— xsd:string ;  
— xsd:integer ;  
— xsd:decimal ;  
— xsd:boolean ;  
— xsd:float ;  
— xsd:double ;  
— xsd:anyURI ;  
— xsd:date ;  
— xsd:time ;  
— xsd:dateTime ;  
— xsd:duration .

**Resource Description Framework (RDF)**

— rdf:type ;  
— rdf:value ;  
— rdf:HTML .

**Simple Knowledge Organization System (SKOS)**

— skos:ConceptScheme ;  
— skos:inScheme ;  
— skos:Concept ;  
— skos:Collection ;

— skos:member ;  
— skos:definition ;  
— skos:notation ;  
— skos:example ;  
— skos:scopeNote ;  
— skos:prefLabel ;  
— skos:altLabel ;  
— skos:editorialNote ;  
— skos:topConceptOf ;  
— skos:narrower ;  
— skos:broader ;  
— skos:exactMatch ;  
— skos:closeMatch ;  
— skos:narrowMatch ;  
— skos:broadMatch ;  
— skos:relatedMatch .

**Resource Description Framework Schema (RDFS)**

— rdfs:Class ;  
— rdfs:Datatype ;  
— rdfs:subClassOf ;  
— rdfs:subPropertyOf ;  
— rdfs:domain ;  
— rdfs:range ;  
— rdfs:label ;  
— rdfs:comment ;  
— rdfs:isDefinedBy ;  
— rdfs:seeAlso ;

## **NTA 8035:2020**

— rdfs:Container ;  
— rdfs:member .

### **Web Ontology Language (OWL)**

— owl:Ontology ;  
— owl:imports ;  
— owl:Class ;  
— owl:DatatypeProperty ;  
— owl:ObjectProperty ;  
— owl:AnnotationProperty ;  
— owl:inverseOf ;  
— owl:FunctionalProperty ;  
— owl:InverseFunctionalProperty ;  
— owl:oneOf ;  
— owl:unionOf ;  
— owl:intersectionOf ;  
— owl:disjointWith ;  
— owl:AllDisjointClasses ;  
— owl:Restriction ;  
— owl:allValuesFrom ;  
— owl:someValuesFrom ;  
— owl:hasValue ;  
— owl:minCardinality ;  
— owl:maxCardinality ;  
— owl:cardinality ;  
— owl:minQualifiedCardinality ;  
— owl:maxQualifiedCardinality ;  
— owl:qualifiedCardinality ;

— owl:onProperty ;  
— owl:onDatatype ;  
— owl:onClass ;  
— owl:Thing ;  
— owl:topDataProperty ;  
— owl:topObjectProperty ;  
— owl:equivalentClass ;  
— owl:equivalentProperty ;  
— owl:sameAs ;  
— owl:hasKey .

**Shape Constraint Language (SHACL): Core & SPARQL-based**

— sh:NodeShape ;  
— sh:PropertyShape ;  
— sh:property ;  
— sh:path ;  
— sh:in ;  
— sh:closed ;  
— sh:ignoredProperties ;  
— sh:qualifiedMinCount ;  
— sh:qualifiedMaxCount ;  
— sh:qualifiedValueShape ;  
— sh:class ;  
— sh:datatype ;  
— sh:targetSubjectsOf ;  
— sh:declare ;  
— sh:message ;  
— sh:prefixes .

**Shape Constraint Language (SHACL): Advanced Features**

- sh:sparql ;
- sh:SPARQLSelect ;
- sh:select ;
- sh:rule ;
- sh:SPARQLRule ;
- sh:construct .

**Bijlage B**  
(normatief)

**Resource Description Format (RDF)**

**Algemeen**

- RDF = Resource Description Framework;
- Ontwikkeld door het World Wide Web Consortium (W3C);
- Een Resource is bij voorkeur een resource op het web, maar kan ook een lokaal bestand zijn;
- RDF is voor machines wat HTML is voor mensen.

**Fundament voor**

- RDFS;
- OWL;
- SHACL (Core en de Advanced Features);
- Serialisaties (verschillende vormen van syntax) zoals:
  - RDF 1.1 XML Syntax;
  - RDF 1.1 Turtle;
  - JSON-LD 1.1 (status: candidate recommendation);
  - SPARQL.

**Gebaseerd op:**

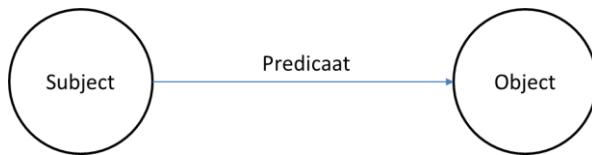
- eenvoudige logica: Predicaten;
- eenvoudige wiskunde: Verzamelingen.

**Logica staat voorop:**

- het data-atoom = een RDF Statement = een Triple:

*Subject – Predicaat – Object.*
- een triple heeft een richting; van links naar rechts;
- een Predicaat staat informeel bekend als Property;
- Property staat in het midden: RDF is ‘property centered’.

**Grafisch**



**Figuur B.1 — Een RDF Statement**

**URI's**

- **Subjecten en Predicaten** gebruiken webadressen (URI's) als identifier:

URI = Uniform Resource Identifier

Als de URI niet alleen als 'identifier' maar ook als 'locator' wordt gebruikt, dan wordt deze dereferenceable genoemd.

- **Objecten** gebruiken URI's of Lexicale waarden:

Een Lexicale waarde is een waarde volgens een bepaald datatype.

Hier is dat bijvoorbeeld een eenvoudig datatype gedefinieerd in XML Schema Part 2: Datatypes, zoals xsd:string, xsd:decimal of xsd:boolean.

**Voorbeelden (let op: deze zijn niet-dereferencable):**

*URI's*

- <http://www.disk.nl/Bridge> ;
- <http://www.history.org/Person> ;
- <http://www.disk.nldesignedBy> ;
- <http://www.material.net/madeOf> ;
- <http://www.disk.nl/span> ;
- <http://www.disk.nl/VanBrienenoordBrug> ;
- <http://www.history.org/designedBy/WJvanderEb> .

*Lexicale waarden*

- "Staal" van het datatype string;
- 287.5 van het datatype decimale waarde;
- true van het datatype boolean.

### Afkortingen

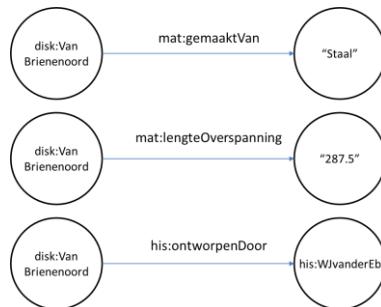
- URI's zijn typisch nogal lang en daarom wat onhandig;
- een zogenoemde prefix kan worden gedefinieerd als afkorting;

VOORBEELD 1 http://www.disk.nl/Brug kan als volgt worden afgekort: prefix disk: <http://www.disk.nl#>  
Dan wordt disk:Brug gebruikt in plaats van http://www.disk.nl/Brug .

VOORBEELD 2 Nog twee voorbeelden van afkortingen:

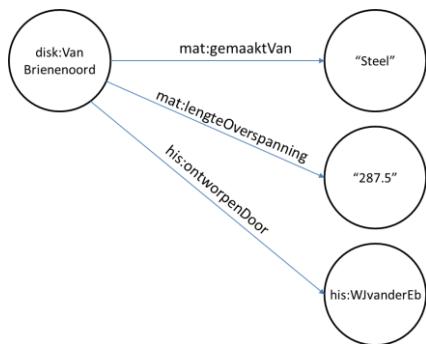
- prefix mat: http://www.material.net# ;
- prefix his: http://www.history.org# .

### Er kunnen drie triples worden gedefinieerd:



Figuur B.2 — Voorbeeld meerdere triples

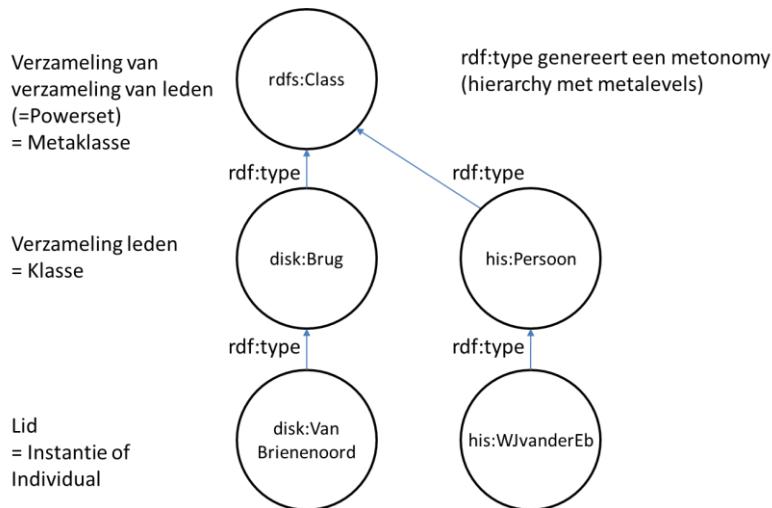
Op een efficiëntere manier:



Figuur B.3 — Triples op een efficiënte manier: een netwerk/graf

Zo ontstaat een netwerk, of nauwkeuriger gezegd: een gerichte, gelabelde graaf.

**Met toevoeging van wat eenvoudige wiskunde:**



**Figuur B.4 — Verzamelingen geven onderscheid instantie en klasse**

**In RDF ziet dat er zo uit:**

(Gebruikmakend van Turtle, een van de standaard formaten (ook wel serialisaties) van RDF die het meest leesbaar is voor mensen).

*Ontologie*

disk:Brug a rdfs:Class ('a' is hierbij een afkorting in Turtle voor 'rdf:type')

mat:gemaaktVan a rdf:Property .

mat:lengteOverspanning a rdf:Property .

his:ontworpenDoor a rdf:Property .

*Data*

disk:VanBrienenoord a disk:Brug ;

(';' is een Turtle-afkorting voor 'hetzelfde subject')

mat:gemaaktVan "Staal"^^xsd:string ;

('^^ geeft het datatype voor de waarde)

mat:lengteOverspanning "287.5"^^xsd:decimal;

his:ontworpenDoor his:WJvanderEb .

his:WJvanderEb a his:Persoon .

OPMERKING 1 De triple is een atomair gegeven. Veranderingen in een graaf betekenen het toevoegen of het verwijderen van een triple, niet het aanpassen van een triple.

OPMERKING 2 Het samenvoegen van grafen betekent het toevoegen van de triples uit de ene graaf aan de andere graaf.

OPMERKING 3 In RDF staan klassen en instanties in hetzelfde bestand.

OPMERKING 4 Het is handig als URI's niet alleen ter identificatie worden gebruikt, maar ook een locatie aangeven. Op deze manier kan iedereen ze makkelijk opzoeken op internet (natuurlijk wel afhankelijk van authenticatie en autorisatie).

#### **Bekend RDF-issue: triple level data**

— De mogelijkheden om 'provenance' (door wie en wanneer is de triple gemaakt of gewijzigd), temporele en ruimtelijke onzekerheden te modelleren op triple-niveau is beperkt:

— De manier om eigenschappen aan triples toe te voegen via het zogenoemde RDF-reificatiemechanisme is omslachtig.

OPMERKING Er moet dubbel worden gemodelleerd (de triple zelf en de gereïficeerde triple): dit is inefficiënt en het ontbreekt aan een standaard manier om de beide triples synchroon met elkaar te houden.

— Gelabelde Property Graphs (LPG's), zoals geïmplementeerd in bijvoorbeeld Neo4j, zijn een betere oplossing.

#### *Workarounds op taalniveau*

— Workaround 1: Reïficatie in de serialisatie (RDF 1.1 N-Quads):

— extra-logical (syntax-only, geen standaard semantiek en gebrek aan software-ondersteuning).

— Workaround 2: Gebruik Named Graphs (NG) (1-triple-grafen):

— NGs for whole graph-mechanisme raakt verloren.

— Workaround 3: Upgrade RDF naar RDF\*, Turtle\* en SPARQL\* die geneste triples ondersteunen:

— nog niet standaard, maar misschien wel in de toekomst als ze W3C Recommendation status krijgen.

#### *Workarounds op modelleerniveau*

— Workaround 4: dupliceer prediciaten als singleton properties die ook weer properties kunnen hebben:

— ten minste RDF/RDFS of OWL Full level;

— inefficiënt (veel class level properties), complex, 'tricky', potentieel inconsistent;

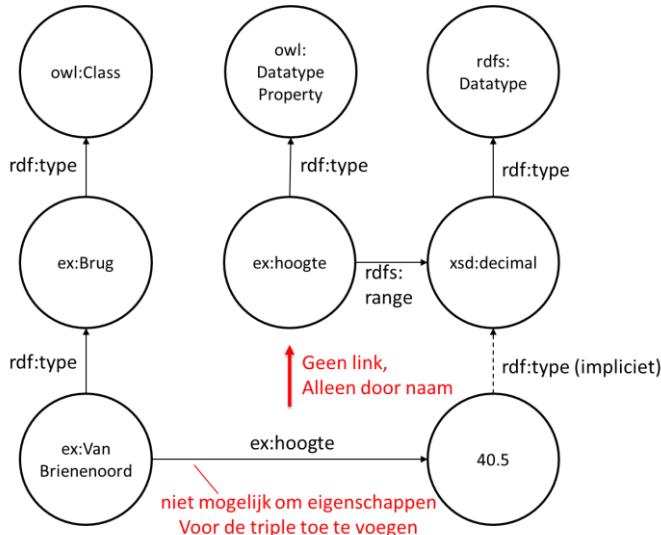
— op deze manier zijn er ook nieuwe metaconstructies nodig (zoals :singletonPropertyOf dat een rdfs:subPropertyOf van rdf:type is), afhankelijk van de gekozen implementatie-aanpak.

— Workaround 5: geobjectificeerde prediciaten (prediciaten worden instanties):

## NTA 8035:2020

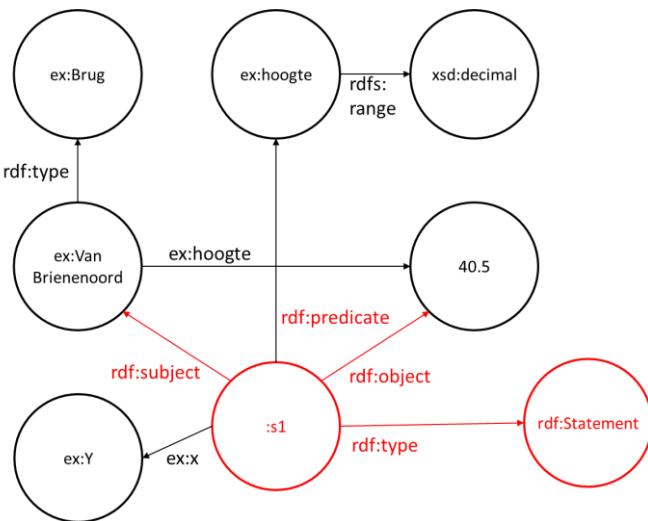
- werkt ook voor minder dan OWL Full;
- indirect, ook hier zijn eigen metaconstructies nodig.

### Voorbeeld om het probleem toe te lichten:



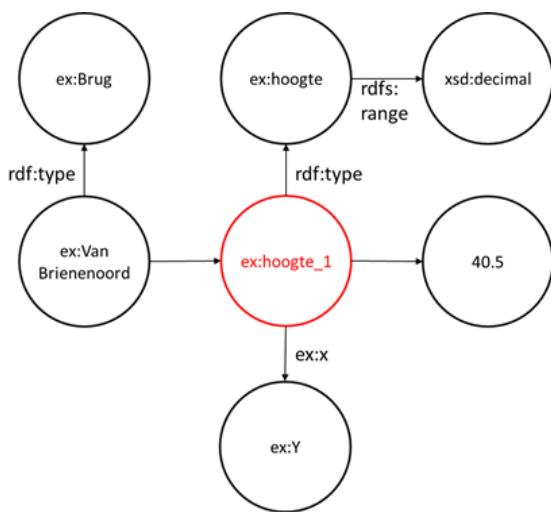
Figuur B.5 — In rood een bekend issue van RDF

### RDF-reificatie:



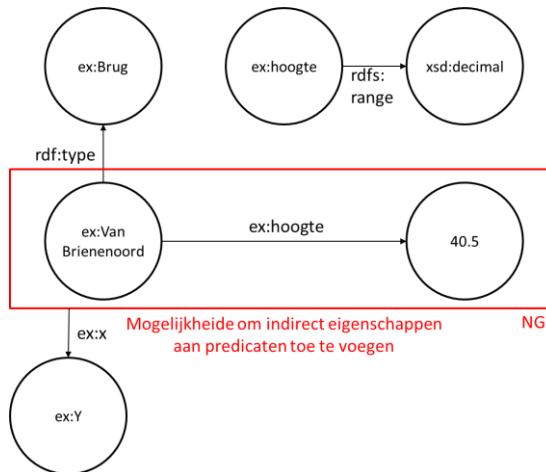
**Figuur B.6 — Standaard RDF-reificatie**

Het gaat beter in Linked Property Graphs (LPG's):

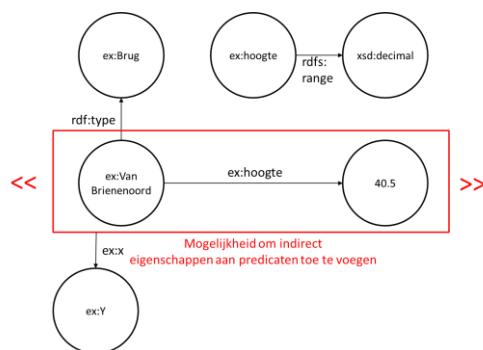
**Figuur B.7 — LPG's kennen wel predicaatinstanties**

Het is mogelijk om eigenschappen voor triples toe te voegen bij instanties van predикатen.

**Gebruik van Names Graphs (NG's):**

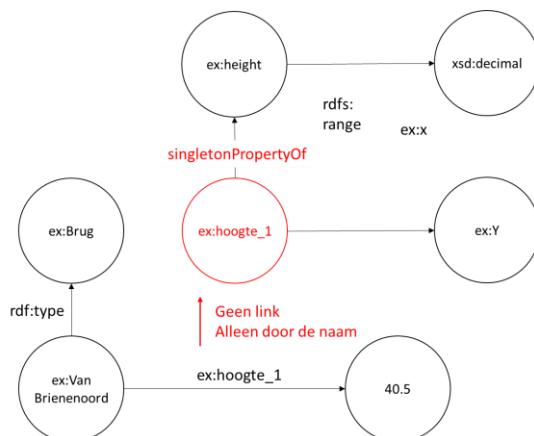
**Figuur B.8 — Gebruik van Names Graphs**

**RDF\*:**

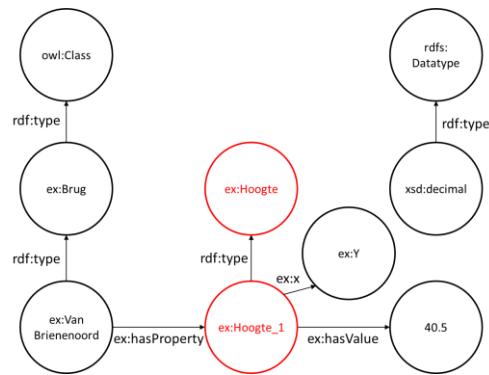


Figuur B.9 — RDF\*, de verwachte opvolger van RDF

**Singleton properties:**



Figuur B.10 — Singleton properties, een andere ~~truuuktruc~~

**Geobjectificeerde predicaten:****Figuur B.11 — Objectificatie van een waarde met bijbehorende klasse**

## Bijlage C (normatief)

### Turtle-formaat

Er bestaan verschillende, in hoge mate gelijkwaardige, concrete formaten (syntaxvormen) voor de serialisatie van RDF, zoals:

- RDF/XML (primaire/formele syntax, op XML gebaseerd);
- N-Triples (niet op XML gebaseerd);
- Turtle (niet op XML gebaseerd, efficiëntere variant van N-Triples);
- JSON-LD (gelijk aan de andere formaten in het geval bepaalde beperkingen op het gebruik van JSON-LD worden toegepast) (niet op XML gebaseerd).

Modelleurs die XML-softwaretools gebruiken hebben vaak een voorkeur voor op XML-gebaseerde formaten. JSON-LD is populair bij ontwikkelaars van op JavaScript gebaseerde semantische webtoepassingen. Het formaat dat de voorkeur heeft voor gebruik in deze NTA is Turtle [Turtle], omdat deze efficiënter is en goed leesbaar, in het bijzonder door het gebruik van zogenoemde voorvoegsels (Engels: prefixes). Het volgende voorbeeld laat dat goed zien.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .  
PREFIX owl: <http://www.w3.org/2002/07/owl#> .  
PREFIX ifc: http://ifcowl.openbimstandards.org/IFC4x1#> .  
ifc:IfcTextTransformation rdf:type owl:Class .
```

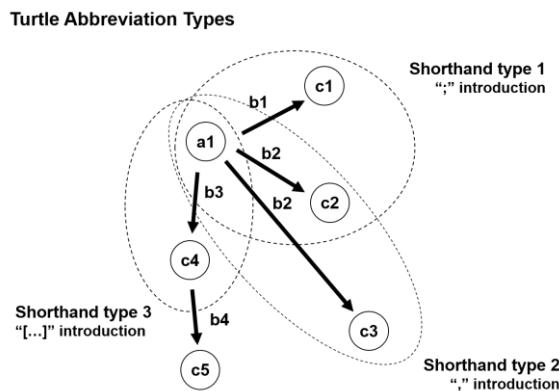
Alle onderwerpen, predicaten en objecten hebben hier een voorvoegsel dat een afkorting is voor een bepaald URI-fragment dat gecombineerd wordt met de werkelijke 'identifier' om de volledige URI te verkrijgen. Merk op dat rdf:type een vooraf gedefinieerd RDF-predicaat is om aan te geven dat een onderwerp van een bepaalde klasse is. Bovendien geeft Turtle de mogelijkheid om nog drie afkortingsmechanismen in te voeren bij het modelleren van een RDF-graaf:

- parallel: zelfde onderwerp via ';' -scheidingsteken, dus één onderwerp met meerdere predicaten;
- parallel: hetzelfde onderwerp-predicaat via ',' scheidingsteken, dus één onderwerp-predicaatcombinatie met meerdere objecten;
- sequentieel: het einde van een triple is het begin van de volgende triple via [...] -groepering.

Om dit te illustreren wordt er uitgegaan van de volgende verzameling triples:

```
a1 b1 c1.  
a1 b2 c2.  
a1 b2 c3.  
a1 b3 c4.  
c4 b4 c5.
```

Vervolgens kunnen alle drie de afkortingstypen worden toegepast zoals weergegeven in figuur C.1.



**Figuur C. 1 — Turtle-formaatafkortingen**

Dit resulteert in de volgende efficiënte code:

```
a1 b1 c1; b2 c2,c3; b3 [c4 b4 c5].
```

Vooral de eerste afkorting (‘;’) wordt heel vaak toegepast wanneer meerdere predicaten voor hetzelfde onderwerp relevant zijn:

```
:RoadSegment_123  
:hasWidth "14.8"^^xsd:float;  
:hasLength "1800.0"^^xsd:float;  
:hasDepth "0.8"^^xsd:float;
```

**NTA 8035:2020**

```
:hasMaterial :OpenAsphaltConcrete-12 ;
:hasExplicitShape :IFC-BoundingBox-321 .
```

Tot slot, de volgende lexicale waarden kunnen worden afgekort:

“tno”^^xsd:string	<>	“tno”
“100”^^xsd:integer	<>	100
“10.5”^^xsd:decimal	<>	10.5

**Bijlage D**  
(normatief)

**CMM en CM in OWL**

Er bestaan twee ontologieën:

- basicsemantics-owl.ttl (benodigd voor simpel modelleren);
- basicsemantics-owl-complex (importeert basicsemantics-owl.ttl) benodigd voor het modelleren van complexe attributen en relaties.

```
# baseURI: https://w3id.org/def/basicsemantics-owl
# imports: http://www.w3.org/2004/02/skos/core
# imports: http://www.w3.org/ns/shacl#
# imports: http://qudt.org/schema/qudt
# imports: http://qudt.org/vocab/unit
# imports: http://qudt.org/vocab/quantitykind
# prefix: bs

@prefix bs: <https://w3id.org/def/basicsemantics-owl#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix quantitykind: <http://qudt.org/vocab/quantitykind/> .
@prefix qudt: <http://qudt.org/schema/qudt/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix unit: <http://qudt.org/vocab/unit/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<https://w3id.org/def/basicsemantics-owl>
  a owl:Ontology ;
  owl:imports <http://www.w3.org/2004/02/skos/core> ;
  owl:imports <http://qudt.org/schema/qudt> ;
```

**NTA 8035:2020**

```
owl:imports <http://qudt.org/vocab/unit> ;
owl:imports <http://qudt.org/vocab/quantitykind> ;

.

bs:Activity
a owl:Class ;
rdfs:subClassOf [
    a owl:Restriction ;
    owl:allValuesFrom bs:Activity ;
    owl:onProperty bs:hasPart ;
] ;
skos:definition "An activity is something possibly or actual happens in space and time" ;
skos:example "Fluid pressure measurement, driving a car are examples of an <activity> " ;
skos:prefLabel "activiteit"@nl ;
skos:prefLabel "activity"@en ;
skos:scopeNote "Process, function, human activity, machine activity, task, procedure, project are subclasses of <Activity>." ;

.

bs:EnumerationType
a owl:Class ;
skos:definition "The superclass of all user-defined enumeration classes where the allowed enumeration items are individuals" ;
skos:example "LoadLevelType being Low, Medium or High" ;
skos:prefLabel "enumeration type"@en ;
skos:prefLabel "enumeratietype"@nl ;

.

bs:Event
a owl:Class ;
skos:definition "A thing that happens or takes place and marks the beginning or ending of an activity or state" ;
```

**NTA 8035:2020**

```
skos:example "The connection of power to pump P_101, the take-off of a aeroplane are examples of an <Event>. " ;  
skos:prefLabel "event"@en ;  
skos:prefLabel "gebeurtenis"@nl ;  
skos:scopeNote "Transition, cause, effect, milestone, issue, accident, point in time are subclasses of <Event>." ;  
. .  
bs:InformationObject  
a owl:Class ;  
rdfs:subClassOf [  
    a owl:Restriction ;  
    owl:allValuesFrom bs:InformationObject ;  
    owl:onProperty bs:hasPart ;  
] ;  
skos:definition "Thing that is a whole of information on itself and has an own identity" ;  
skos:example "a file within a computer system, the PDF file with URI http://material-certificate/M-101-1234.pdf are examples of <InformationObject> " ;  
skos:prefLabel "informatieobject"@nl ;  
skos:prefLabel "information object"@en ;  
skos:scopeNote "Requirement, document, statement, E-mail, code, identifier are subclasses of <InformationObject> ." ;  
. .  
bs:PhysicalObject  
a owl:Class ;  
rdfs:subClassOf [  
    a owl:Restriction ;  
    owl:allValuesFrom bs:PhysicalObject ;  
    owl:onProperty bs:hasPart ;  
] ;
```

## NTA 8035:2020

```
skos:definition "Is something that possibly or actual exists in space  
and time, perceptible through the senses" ;  
  
skos:example "Pump P_101, a single living organism are examples of a  
<PhysicalObject>" ;  
  
skos:prefLabel "fysiek object"@nl ;  
skos:prefLabel "physical object"@en ;  
  
skos:scopeNote "Artefact, matter, person, organisation, stream,  
connection are subclasses of <PhysicalObject>." ;  
  
.  
  
bs:SpatialRegion  
  
a owl:Class ;  
  
skos:definition "Demarcated space" ;  
  
skos:example "construction area for a building, maritime traffic zone  
within the Channel, a hazard zone within a plant and the loading gauge of  
a train are examples of a <SpatialRegion>" ;  
  
skos:prefLabel "ruimtelijk gebied"@nl ;  
skos:prefLabel "spatial region"@en ;  
  
. .  
  
bs:State  
  
a owl:Class ;  
  
skos:definition "A particular condition that something is in during a  
specific period in time" ;  
  
skos:example "well-functioning of a car, the condition of an elevator  
of being un-safe are examples of a <State>" ;  
  
skos:prefLabel "state"@en ;  
skos:prefLabel "toestand"@nl ;  
  
skos:scopeNote "Condition, risk, failure state, objective are  
subclasses of <State>." ;  
  
. .  
  
bs:TemporalRegion  
  
a owl:Class ;  
skos:definition "A length or portion of time" ;
```

**NTA 8035:2020**

```
skos:example "week 12 in 2019, design stage of the Schiphol tunnel are
examples of a <TemporalRegion>" ;

skos:prefLabel "temporeel gebied"@nl ;
skos:prefLabel "temporal region"@en ;

skos:scopeNote "Week, day, life-cycle stage are subclasses of
<TemporalRegion>." ;

.

bs:begins

a owl:ObjectProperty ;
skos:prefLabel "begins"@en ;
skos:prefLabel "begint"@nl ;

.

bs:describes

a owl:ObjectProperty ;
skos:prefLabel "beschrijft"@nl ;
skos:prefLabel "describes"@en ;

.

bs:ends

a owl:ObjectProperty ;
skos:prefLabel "beeindigt"@nl ;
skos:prefLabel "ends"@en ;

.

bs:hasBoundary

a owl:ObjectProperty ;
skos:prefLabel "has boundary"@en ;
skos:prefLabel "heeft begrenzing"@nl ;

.

bs:hasInterior

a owl:ObjectProperty ;
skos:prefLabel "has interior"@en ;
```

**NTA 8035:2020**

```
skos:prefLabel "heeft inwendige"@nl ;  
.  
bs:hasPart  
a owl:ObjectProperty ;  
skos:prefLabel "has part"@en ;  
skos:prefLabel "heeft deel"@nl ;  
.br/>bs:hasPeriod  
a owl:ObjectProperty ;  
skos:prefLabel "has period"@en ;  
skos:prefLabel "heeft periode"@nl ;  
.br/>bs:hasState  
a owl:ObjectProperty ;  
skos:prefLabel "has state"@en ;  
skos:prefLabel "heeft toestand"@nl ;  
.br/>bs:abbreviation  
a owl:DatatypeProperty ;  
rdfs:subPropertyOf skos:altLabel ;  
skos:prefLabel "abbreviation"@en ;  
skos:prefLabel "afkorting"@nl ;  
.br/>bs:performs  
a owl:ObjectProperty ;  
skos:prefLabel "performs"@en ;  
skos:prefLabel "voert uit"@nl ;  
.br/>bs:transforms
```

**NTA 8035:2020**

```
a owl:ObjectProperty ;
skos:prefLabel "transformeert"@nl ;
skos:prefLabel "transforms"@en ;

.

bs:triggers
a owl:ObjectProperty ;
skos:prefLabel "triggers"@en ;
skos:prefLabel "triggert"@nl ;

.

[

a owl:AllDisjointClasses ;
owl:members (
    bs:PhysicalObject
    bs:InformationObject
    bs:State
    bs:Event
    bs:TemporalRegion
    bs:SpatialRegion
    bs:Activity
) ;
].

bs:unit
a owl:ObjectProperty ;
skos:prefLabel "unit"@en ;
skos:prefLabel "eenheid"@nl ;

.

bs:quantityKind
a owl:ObjectProperty ;
skos:prefLabel "quantity kind"@en ;
```

**NTA 8035:2020**

```
skos:prefLabel "grootheid"@nl ;  
.  
bs:hasSpatialLocation  
a owl:ObjectProperty ;  
skos:prefLabel "has spatial location"@en ;  
skos:prefLabel "heeft ruimtelocatie"@nl ;  
.  
bs:SpatialLocation  
a owl:Class ;  
skos:prefLabel "spatial location"@en ;  
skos:prefLabel "ruimtelocatie"@nl ;  
.  
bs:hasSpatialReferenceSystem  
a owl:ObjectProperty ;  
skos:prefLabel "has spatial reference system"@en ;  
skos:prefLabel "heeft ruimterefereniesysteem"@nl ;  
.  
bs:hasSpatialReference  
a owl:ObjectProperty ;  
skos:prefLabel "has spatial reference"@en ;  
skos:prefLabel "heeft ruimtereferentie"@nl ;  
.  
bs:SpatialReferenceSystem  
a owl:Class ;  
skos:prefLabel "spatial reference system"@en ;  
skos:prefLabel "ruimterefereniesysteem"@nl ;  
.  
bs:SpatialReference  
a owl:Class ;
```

```
skos:prefLabel "spatial reference"@en ;
skos:prefLabel "ruimtereferentie"@nl ;

.

bs:AddressSystem

a owl:Class ;
rdfs:subClassOf bs:SpatialReferenceSystem ;
skos:prefLabel "address system"@en ;
skos:prefLabel "adressysteem"@nl ;

.

bs:CoordinateReferenceSystem

a owl:Class ;
rdfs:subClassOf bs:SpatialReferenceSystem ;
skos:prefLabel "coordinate reference system"@en ;
skos:prefLabel "coordinaatreferentiesysteem"@nl ;

.

bs:IndexKnownSpatialLocations

a owl:Class ;
rdfs:subClassOf bs:SpatialReferenceSystem ;
skos:prefLabel "index known spatial locations"@en ;
skos:prefLabel "index bekende ruimtelijke plaatsen"@nl ;

.

bs:LinearReferenceSystem

a owl:Class ;
rdfs:subClassOf bs:SpatialReferenceSystem ;
skos:prefLabel "linear reference system"@en ;
skos:prefLabel "lineair referentiesysteem"@nl ;

.

bs:hasTemporalLocation

a owl:ObjectProperty ;
```

**NTA 8035:2020**

```
skos:prefLabel "has temporal location"@en ;
skos:prefLabel "heeft tijdlocatie"@nl ;

.

bs:TemporalLocation
skos:prefLabel "temporal location"@en ;
skos:prefLabel "tijdlocatie"@nl ;

.

bs:hasTemporalReferenceSystem
a owl:ObjectProperty ;
skos:prefLabel "has temporal reference system"@en ;
skos:prefLabel "heeft tijddreferentiesysteem"@nl ;

.

bs:hasTemporalReference
a owl:ObjectProperty ;
skos:prefLabel "has temporal reference"@en ;
skos:prefLabel "heeft tijddreferentie"@nl ;

.

bs:TemporalReferenceSystem
a owl:Class ;
skos:prefLabel "temporal reference system"@en ;
skos:prefLabel "tijddreferentiesysteem"@nl ;

.

bs:TemporalReference
a owl:Class ;
skos:prefLabel "temporal reference"@en ;
skos:prefLabel "tijddreferentie"@nl ;

.

bs:TemporalCoordinateReferenceSystem
a owl:Class ;
```

**NTA 8035:2020**

```
rdfs:subClassOf bs:TemporalReferenceSystem ;
skos:prefLabel "temporal coordinate reference system"@en ;
skos:prefLabel "tijdcoördinaatreferentiesysteem"@nl ;

.

bs:CalenderSystem
a owl:Class ;
rdfs:subClassOf bs:TemporalReferenceSystem ;
skos:prefLabel "calender system"@en ;
skos:prefLabel "kalendersysteem"@nl ;
.
```

**NTA 8035:2020**

```
# baseURI: https://w3id.org/def/basicsemantics-owl-complex
# imports: https://w3id.org/def/basicsemantics-owl
# prefix: bsc

@prefix bs: <https://w3id.org/def/basicsemantics-owl#> .
@prefix bsc: <https://w3id.org/def/basicsemantics-owl-complex#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix quantitykind: <http://qudt.org/vocab/quantitykind/> .
@prefix qudt: <http://qudt.org/schema/qudt/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix unit: <http://qudt.org/vocab/unit/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<https://w3id.org/def/basicsemantics-owl-complex>
  a owl:Ontology ;
  owl:imports <https://w3id.org/def/basicsemantics-owl> ;
  .

  bsc:Property
    a owl:Class ;
    skos:prefLabel "aspect: attribuut of relatie"@nl ;
    skos:prefLabel "property"@en ;
  .

  bsc:hasProperty
    a owl:ObjectProperty ;
    skos:prefLabel "has property"@en ;
    skos:prefLabel "heeft aspect"@nl ;
  .
```

```
bsc:hasPropertySet
a owl:ObjectProperty ;
skos:prefLabel "has property group"@en ;
skos:prefLabel "heeft attribuut/relatie groep"@nl ;
.

bsc:hasPropertyDef
a owl:ObjectProperty ;
skos:prefLabel "has property definition"@en ;
skos:prefLabel "heeft attribuut/relatie definitie"@nl ;
.
```

**Bijlage E**  
(normatief)

**Modelleerstappen**

**Voor Level of Capability 1 (SKOS)**

- 1) Declareer prefixen & skos:ConceptScheme.
- 2) Declareer skos:Concepten (deze vertegenwoordigen hier zowel dingen op conceptniveau als op instantieniveau).
- 3) Verbind de concepten via skos:related, in het bijzonder via skos:broader of skos:narrower:
  - Verbind eigen skos:Concepten via skos:broader met top level-concepten.
- 4) Geef aan welk skos:Concept hoort bij welk SKOS-schema via skos:inScheme.
- 5) Voeg (eventueel meertalige) annotaties toe voor de resultaten van 1) t/m 3).

**Voor Level of Capability 2 (RDFS)**

- 1) Declareer prefixes & owl:Ontology via namespace URI + imports.
- 2) Declareer rdfs:Class'en voor de Concepten.
- 3) Declareer rdfs:Class'en voor Enumeration Datatypes en definieer de toegestane enumeration items als (referentie-)instanties.
- 4) Declareer rdf:Properties, inclusief varianten met kleine letters voor alle klassen uit 3).
- 5) Definieer rdfs:subClassOf-relaties tussen de klassen uit 2), resulterend in een taxonomie & rdfs:subPropertyOf-relaties tussen de eigenschappen uit 4):
  - Maak ze subclass/subproperty van de top level classes/properties.
- 6) Voeg optioneel domein en bereik toe voor de eigenschappen uit 4) als property-beperkingen.
- 7) Voeg optioneel (eventueel meertalige) annotaties toe voor de classes/properties uit 2) t/m 4).
- 8) Definieer Individuals (inclusief property-waarden) en typeer ze met behulp van rdf:type naar de Concepten uit 2).
- 9) Voeg optioneel (eventueel meertalige) annotaties toe voor de Individuals uit 8).

**Voor Level of Capability 3a (OWL)**

- 1) Declareer prefixes & owl:Ontology via namespace URI + imports.
- 2) Declareer owl:Class'en voor de Concepten.
- 3) Declareer owl:Class'en voor Enumeration Datatypes en definieer de toegelaten enumeratie-items als (referentie-)instanties, inclusief een owl:oneOf-beperking in het geval van een gesloten lijst.
- 4) Declareer owl:DatatypeProperties voor attributen.
- 5) Declareer owl:ObjectProperties voor relaties:
  - inclusief varianten met kleine letters voor alle klassen uit 3);
  - inclusief inverse object properties als nodig/relevant.
- 6) Definieer rdfs:subClassOf-relaties tussen de klassen uit 2) resulterend in een taxonomie & rdfs:subPropertyOf-relaties tussen de eigenschappen uit 4) en 5):
  - Maak ze subclass/subproperty van de top level classes en properties.
- 7) Voeg optioneel domein en bereik toe voor de eigenschappen uit 4) en 5) als property-beperkingen.
- 8) Voeg optioneel (eventueel meertalige) annotaties toe voor de klassen en properties uit 2) t/m 5).
- 9) Definieer OWA beperkingen in OWL for Classes:
  - inclusief beperkingen op relaties zoals decompositie (bs:hasPart). Op deze manier wordt een meronomie verkregen.
  - inclusief beperkingen op inverse object-relaties indien relevant.
- 10) Definieer Individuals (inclusief property-waarden) en typeer deze met behulp van rdf:type naar de Concepten uit 2).
- 11) Voeg optioneel (eventueel meertalige) annotaties toe voor de Individuals uit 10).

**Voor Level of Capability 3b (SHACL)**

- 1) Declareer prefixes & owl:Ontology via namespace URI + imports.
- 2) Declareer sh:NodeShape's voor de Concepten.
- 3) Declareer owl:Class'en voor Enumeration Datatypes en definieer de toegelaten enumeratie-items als (referentie-)instanties, inclusief een sh:in-beperking in het geval van een gesloten lijst.
- 4) Declareer (eventueel) owl:DatatypeProperties.
- 5) Declareer (eventueel) owl:ObjectProperties:

## NTA 8035:2020

- inclusief varianten met kleine letters voor alle klassen uit 3);
  - inclusief inverse object properties als nodig/relevant.
- 6) Definieer rdfs:subClassOf-relaties tussen de klassen uit 2) resulterend in een taxonomie & rdfs:subPropertyOf-relaties tussen de eigenschappen uit 4) en 5):
- Maak ze subclass/subproperty van de top level classes en properties.
- 7) Voeg optioneel (eventueel meertalige) annotaties toe voor de klassen en properties uit 2) t/m 5).
- 8) Definieer (of genereer uit de OWL OWA constraints) CWA SHACL shapes:
- a) inclusief: definieer sh:PropertyShape's voor rdfs:range's;
  - b) in het algemeen, zie "SHACL and OWL Compared" [via de URL http://spinrdf.org/shacl-and-owl.html](http://spinrdf.org/shacl-and-owl.html);
  - c) geavanceerd: voeg afleidingsregels toe volgens SHACL-AF.

**Met opmerkingen [BH(22):** via biblio link?

**Met opmerkingen [BH(23R22):** wim/tom, kunnen jullie dit doen?

**Bijlage F**  
(normatief)

**Voorbeeld van simpele Attributen en Relaties in OWL**

```
# baseURI: https://w3id.org/def/example1
# imports: https://w3id.org/def/basicsemantics-owl
# prefix: ex1

@prefix bs: <https://w3id.org/def/basicsemantics-owl#> .
@prefix ex1: <https://w3id.org/def/example1#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix quantitykind: <http://qudt.org/vocab/quantitykind/> .
@prefix qudt: <http://qudt.org/schema/qudt/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix unit: <http://qudt.org/vocab/unit/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<https://w3id.org/def/example1>
  a owl:Ontology ;
  owl:imports <https://w3id.org/def/basicsemantics-owl> ;

  ex1:Bridge
    a owl:Class ;
    rdfs:label "Bridge"@en ;
    rdfs:label "Brug"@nl ;
    rdfs:subClassOf bs:PhysicalObject ;
    rdfs:subClassOf [
```

**NTA 8035:2020**

```
a owl:Restriction ;
owl:cardinality "1"^^xsd:nonNegativeInteger ;
owl:onProperty ex1:height ;
] ;
rdfs:subClassOf [
a owl:Restriction ;
owl:onClass ex1:Deck ;
owl:onProperty bs:hasPart ;
owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger ;
] ;
skos:definition "Verbinding voor verkeer tussen twee landhoofden die gescheiden wordt door water."@nl ;

.

ex1:Bridge_1
a ex1:Bridge ;
bs:hasPart ex1:Deck_1 ;
ex1:currentlyServingVehicle ex1:Vehicle_1 ;
ex1:height 50.0 ;

.

ex1:Deck
a owl:Class ;
rdfs:subClassOf bs:PhysicalObject ;
rdfs:subClassOf [
a owl:Restriction ;
owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
owl:onClass ex1:Slab ;
owl:onProperty bs:hasPart ;
] ;
.
```

```
ex1:Deck_1
  a ex1:Deck ;
  bs:hasPart ex1:Slab_1 ;
  bs:hasPart ex1:Slab_2 ;
  bs:hasPart ex1:Slab_3 ;

.

ex1:Heavy
  a ex1:LoadLevelType ;

.

ex1:Light
  a ex1:LoadLevelType ;

.

ex1:LoadLevelType
  a owl:Class ;
  rdfs:subClassOf bs:EnumerationType ;
  owl:equivalentClass [
    a owl:Class ;
    owl:oneOf (
      ex1:Light
      ex1:Normal
      ex1:Heavy
    ) ;
  ] ;

.

ex1:Normal
  a ex1:LoadLevelType ;

.

ex1:Slab
  a owl:Class ;
```

**NTA 8035:2020**

```
rdfs:subClassOf bs:PhysicalObject ;  
.  
ex1:Slab_1  
a ex1:Slab ;  
. .  
ex1:Slab_2  
a ex1:Slab ;  
. .  
ex1:Slab_3  
a ex1:Slab ;  
. .  
ex1:Vehicle  
a owl:Class ;  
rdfs:subClassOf bs:PhysicalObject ;  
. .  
ex1:Vehicle_1  
a ex1:Vehicle ;  
ex1:loadLevel ex1:Heavy ;  
ex1:velocity 128.0 ;  
. .  
ex1:currentlyServingVehicle  
a owl:ObjectProperty ;  
. .  
ex1:height  
a owl:DatatypeProperty ;  
rdfs:range xsd:decimal ;  
bs:quantityKind quantitykind:Length ;  
bs:unit unit:M ;  
. .
```

**NTA 8035:2020**

```
ex1:loadLevel
  a owl:ObjectProperty ;
  .

ex1:velocity
  a owl:DatatypeProperty ;
  bs:quantityKind quantitykind:Speed ;
  bs:unit unit:KiloM-PER-HR ;
  .
```

## Bijlage G (normatief)

### Complexe properties

Properties worden hier bedoeld in de zin van RDF Property (predicaat); het gaat dus om zowel complexe attributen als complexe relaties.

#### Geïnspireerd door de bestaande ontologie: OPM

- Ontology for Property Management (OPM):
  - gebaseerd op het paper *OPM, An Ontology for describing properties that evolve over time* [22].
- Context:
  - World Wide Web Consortium (W3C):
    - Linked Building Data (LBD) Community Group (CG).

#### OPM definieert twee soorten properties:

- 1) numerieke datatype-properties, zoals 'hoogte';
- 2) lexicale datatype-properties, zoals 'materiaal'.

Het gaat hier in het kort dus om kwantiteiten en kwaliteiten.

VOORBEELD 1 ‘Hoogte’ heeft een waarde die wordt uitgedrukt in een schaal met een bepaalde eenheid en die gerelateerd is aan een basis of afgeleide groothed.

VOORBEELD 2 ‘Materiaal’ heeft een verwijzing naar een bepaalde referentie-instantie als toegelaten enumeratie-item.

#### Drie OPM-complexiteitsniveaus:

- Niveau L1: simpel, geen objectificatie van properties;
- Niveau L2: complex: eenmalige objectificatie;
- Niveau L3: nog complexer: dubbele objectificatie.

Hierbij moet het volgende in overweging worden genomen:

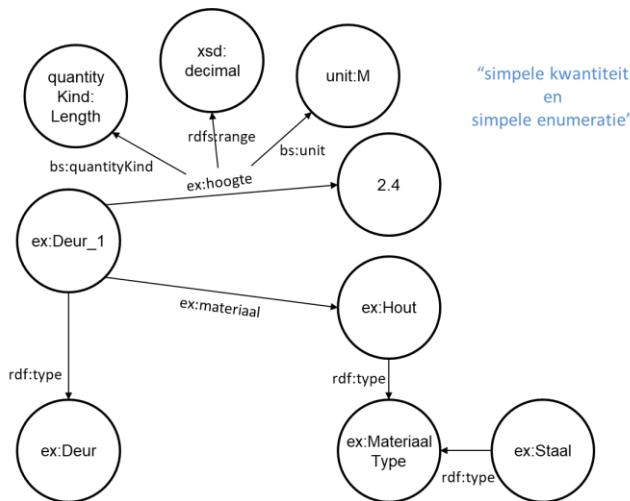
- Er kan één niveau worden gekozen, maar er kunnen ook meerdere niveaus worden gecombineerd.
- L1 kan uit L2 worden afgeleid, hierbij treedt wel dataverlies op (idem. voor L2 uit L3).
- Er zijn verschillende mogelijkheden om eventueel gecombineerde niveaus ‘in sync’ te houden (bijvoorbeeld via standaard SPARQL queries).
- OPM past altijd CDT/UCUM voor grootheden en eenheden toe:
  - Waarden met eenheid worden genoteerd als zogenoemde Well Known Text (WKT) strings, zoals 2.4 m;

- Grootheden worden gebruikt als onderliggende datatypen.

**In deze NTA wordt de volgende werkwijze afgesproken:**

- De prefixen die worden gebruikt: bs & bsc (bsc bij complex properties).
- Alleen OPM-niveaus L1 (simpel) en L2 (complex) worden gebruikt.
- Simpele kwaliteiten zijn niet noodzakelijkerwijs opgenomen in een enumeratie (er zijn ook niet-enumeratiekwaliteiten zoals van het type xsd:string en xsd:boolean).
- In plaats van schema:value wordt rdf:value gehanteerd (de schema-variant heeft een te specifieke invulling). De rdfs:range van rdf:value is een rdfs:Resource dus ook instantie-waarden worden hiermee gedekt bij relaties.
- Een complexe property instantie (bijvoorbeeld Hoogte\_123) is een instantie van bsc:Property en verwijst voor zijn definitie via bsc:hasPropertyDef naar een simpele variant (height owl:DatatypeProperty).
- Er wordt een groeperingmechanisme toegevoegd via rdfs:Container en rdfs:member. Dit mechanisme kan gebruikt worden om zowel property definities als complexe instanties te groeperen. Een groep instanties kan toegewezen worden aan een object via bsc:hasPropertySet naast de bestaande bsc:hasProperty.
- Consistent wordt QUDT (versie 2.1) gebruikt voor grootheden en eenheden in plaats van CDT/UCUM van OPM.
- Het modelleerpatroon wordt ook toegepast op relaties (inclusief enumeratie-attributen die als relaties zijn gemodelleerd).
- Merk op dat instanties van complexe properties anoniem kunnen zijn. In de volgende voorbeelden in bijlagen H, I en J hebben ze voor de duidelijkheid echter wel een naam gekregen (zoals bijvoorbeeld Hoogte\_1).

**Level L1 (simpel attribuut en simpele enumeratie)**



Figuur G.1 — Simpele modelleren, grafisch

**In OWL/Turtle**

**Ontology**

```

ex:Deur rdf:type owl:Class .

ex:MateriaalType rdf:type owl:Class .

ex:Hout rdf:type ex:MateriaalType .

ex:Staal rdf:type ex:MateriaalType .

ex:hoogte rdf:type owl:DatatypeProperty

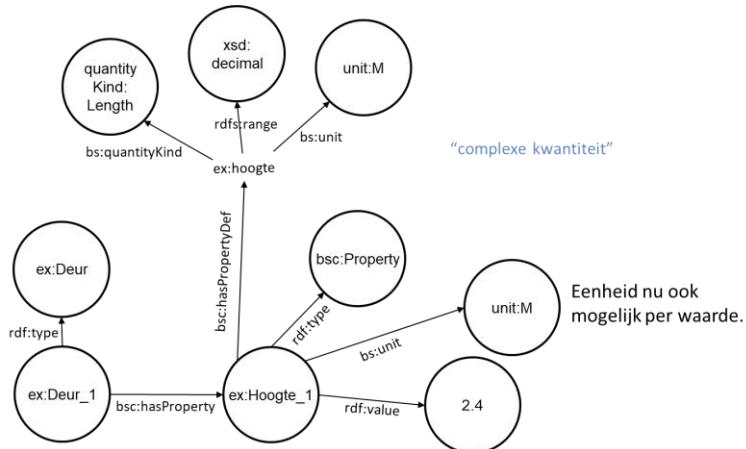
    rdfs:range xsd:decimal ;
    bs:unit unit:M ;
    bsc:quantityKind quantitykind:Length .

ex:materiaal rdf:type owl:ObjectProperty .
  
```

**Data**

```

ex:Deur_1 rdf:type ex:Deur ;
    ex:hoogte 2.40 ;
    ex:materiaal ex:Hout .
  
```

**Level L2 (complex attribuut, hier kwantiteit)****Figuur G.2 — Complex modellering van een complexe kwantiteit, grafisch****In OWL/Turtle****Ontology**

```

bsc:Property rdf:type owl:Class .
bsc:hasProperty rdf:type owl:ObjectProperty .
bsc:hasPropertyDef rdf:type owl:ObjectProperty .

bs:unit rdf:type owl:ObjectProperty .

bs:quantityKind rdf:type owl:ObjectProperty .

ex:Deur rdf:type owl:Class .

ex:hoogte rdf:type owl:DatatypeProperty ;
    rdfs:range xsd:decimal ;
    bs:unit unit:M ;
    bsc:quantityKind quantitykind:Length .
  
```

**Data**

```

ex:Deur_1 rdf:type ex:Deur ;
    bsc:hasProperty ex:Hoogte_1 .

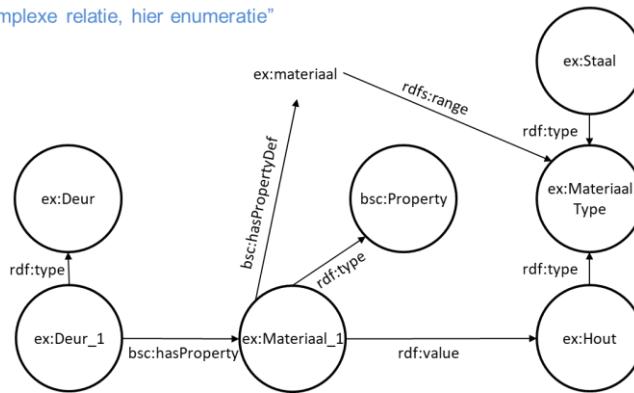
ex:Hoogte_1 rdf:type bsc:Property ;
  
```

## NTA 8035:2020

```
bsc:hasPropertyDef ex:hoogte ;  
bsc:unit unit:M ; -- het is nu ook mogelijk om eenheid per waarde te geven  
rdf:value 2.4 .
```

### Level L2 (complexe relatie, hier naar een enumeratie)

"complexe relatie, hier enumeratie"



Figuur G.3 — Modelleren van een complexe relatie, grafisch

### Level L2 – In OWL/Turtle

#### Ontology

```
bsc:Property rdf:type owl:Class .  
bsc:hasProperty rdf:type owl:ObjectProperty .  
bsc:hasPropertyDef rdf:type owl:ObjectProperty .  
ex:Deur rdf:type owl:Class .  
ex:MaterialType rdf:type owl:Class .  
ex:Hout rdf:type ex:MaterialType .  
ex:Staal rdf:type ex:MaterialType .  
ex:materiaal rdf:type owl:ObjectProperty .
```

#### Data

```
ex:Deur_1 rdf:type ex:Deur ;  
    bsc:hasProperty ex:Materiaal_1 .  
ex:Materiaal_1 rdf:type bsc:Property ;
```

```
bsc:hasPropertyDef ex:materiaal ;  
rdf:value ex:Hout .
```

#### **Toekomstige acties**

- Combinatie met een eventueel productmodelleerpatroon.
- Afstemming internationaal.
- W3C Linked Building Data (LBD) Community Group;
- CEN TC442/WG4 dealing with Product Data Templates (PDT's);
- bSI Technical Room/Product Room dealing with IFC, property sets en bSDD.

**Bijlage H**  
(normatief)

**Voorbeeld van complexe Attributen in OWL**

```
# baseURI: https://w3id.org/def/example2
# imports: https://w3id.org/def/basicsemantics-owl-complex
# prefix: ex2

@prefix bs: <https://w3id.org/def/basicsemantics-owl#> .
@prefix bsc: <https://w3id.org/def/basicsemantics-owl-complex#> .
@prefix ex2: <https://w3id.org/def/example2#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix quantitykind: <http://qudt.org/vocab/quantitykind/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix unit: <http://qudt.org/vocab/unit/> .
@prefix qudt: <http://qudt.org/schema/qudt/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<https://w3id.org/def/example2>
  a owl:Ontology ;
  owl:imports <https://w3id.org/def/basicsemantics-owl-complex> ;

  ex2:Door
    a owl:Class ;
    rdfs:subClassOf bs:PhysicalObject ;

  ex2:Door_1
```

```
a ex2:Door ;
bsc:hasProperty ex2:ClearOpeningHeight_1 ;

.
ex2:height
a owl:DatatypeProperty ;
rdfs:range xsd:decimal ;
bs:unit unit:M ;
bs:quantityKind quantitykind:Length ;

.
ex2:clearOpeningHeight
a owl:DatatypeProperty ;
rdfs:subPropertyOf ex2:height ;
rdfs:seeAlso "EN12519" ;
rdfs:member ex2:Geometric-properties ;

.
ex2:ClearOpeningHeight_1
a bsc:Property ;
rdf:value 2.40 ;
bsc:hasPropertyDef ex2:clearOpeningHeight ;
ex2:measuredBy "Someone" ;

.
ex2:GeometricProperties
a rdfs:Container ;

.
ex2:measuredBy
a owl:DatatypeProperty ;
rdfs:range xsd:string ;
```

**Bijlage I**  
(normatief)

**Voorbeeld van complexe Relaties in OWL**

```
# baseURI: https://w3id.org/def/example3
# imports: https://w3id.org/def/basicsemantics-owl-complex
# prefix: ex3

@prefix bs: <https://w3id.org/def/basicsemantics-owl#> .
@prefix bsc: <https://w3id.org/def/basicsemantics-owl-complex#> .
@prefix ex3: <https://w3id.org/def/example3#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix quantitykind: <http://qudt.org/vocab/quantitykind/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix unit: <http://qudt.org/vocab/unit/> .
@prefix qudt: <http://qudt.org/schema/qudt/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<https://w3id.org/def/example3>
  a owl:Ontology ;
  owl:imports <https://w3id.org/def/basicsemantics-owl-complex> ;

  ex3:Bridge
    a owl:Class ;
    rdfs:subClassOf bs:PhysicalObject ;

  ex3:Bridge_1
```

```
a ex3:Bridge ;
bs:hasProperty ex3:CurrentlyServingVehicle_1 ;

.

ex3:currentlyServingVehicle
a owl:ObjectProperty ;

.

ex3:CurrentlyServingVehicle_1
a bsc:Property ;
rdf:value ex3:Vehicle_1 ;
ex3:measuredBy "Sensor_100" ;
bsc:hasPropertyDef ex3:currentlyServingVehicle ;

.

ex3:Vehicle
a owl:Class ;
rdfs:subClassOf bs:PhysicalObject ;

.

ex3:Vehicle_1
a ex3:Vehicle ;

.

ex3:measuredBy
a owl:DatatypeProperty ;
rdfs:range xsd:string ;
```

**Bijlage J**  
(normatief)

**Voorbeeld van complexe Relaties voor enumeraties in OWL**

```
# baseURI: https://w3id.org/def/example4
# imports: https://w3id.org/def/basicsemantics-owl-complex
# prefix: ex4

@prefix bs: <https://w3id.org/def/basicsemantics-owl#> .
@prefix bsc: <https://w3id.org/def/basicsemantics-owl-complex#> .
@prefix ex4: <https://w3id.org/def/example4#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix quantitykind: <http://qudt.org/vocab/quantitykind/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix unit: <http://qudt.org/vocab/unit/#> .
@prefix qudt: <http://qudt.org/schema/qudt/#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<https://w3id.org/def/example4>
  a owl:Ontology ;
  owl:imports <https://w3id.org/def/basicsemantics-owl-complex> ;

  ex4:Vehicle
    rdf:type owl:Class ;
    rdfs:subClassOf bs:PhysicalObject ;

  ex4:Vehicle_1
```

```
rdf:type ex4:Vehicle ;
bs:hasProperty ex4:LoadLevel_1 ;

.

ex4:LoadLevelType
a owl:Class ;
rdfs:subClassOf bs:EnumerationType ;
owl:equivalentClass [
rdf:type owl:Class ;
owl:oneOf (ex4:Light ex4:Normal ex4:Heavy) ] ;

.

ex4:Light
rdf:type ex4:LoadLevelType ;

.

ex4:Normal
rdf:type ex4:LoadLevelType ;

.

ex4:Heavy
rdf:type ex4:LoadLevelType ;

.

ex4:loadLevel
rdf:type owl:ObjectProperty;
ex4:status "final" ;

.

ex4:LoadLevel_1
rdf:type bsc:Property ;
rdf:value ex4:Heavy;
ex4:certainty 0.5 ;
bsc:hasPropertyDef ex4:loadLevel ;

.
```

**NTA 8035:2020**

```
ex4:certainity
a owl:DatatypeProperty ;
rdfs:range xsd:decimal ;

.
.
.
ex4:status
a owl:DatatypeProperty ;
rdfs:range xsd:string ;
```

## Bibliografie

NEN 2767:reeks, *Conditiemeting gebouwde omgeving, ...*

NEN 2082, *Eisen voor functionaliteit van informatie- en archiefmanagement in programmatuur*

ISO 10646, *Information technology – Universal Coded Character Set (UCS)*

NEN-EN-ISO 19136-2:2018 (en) Geographic information - Geography Markup Language (GML) - Part 2: Extended schemas and encoding rules

ISO/IEC 11578, *Information technology – Open Systems Interconnection – Remote Procedure Call (RPC)*

[1] Terry Halpin, Tony Morgan; *Information Modelling and Relational Databases . The Morgan Kaufmann Series in Data Management Systems* 2<sup>nd</sup> Edition.

[2] [https://ris.utwente.nl/ws/portalfiles/portal/6042428/thesis\\_Guizzardi.pdf](https://ris.utwente.nl/ws/portalfiles/portal/6042428/thesis_Guizzardi.pdf).

[3] ISO 21597, Information Container for Data Delivery (ICDD), Deel 1 "container met gelinkte documenten" en Deel 2 "extra semantiek voor links" (in voorbereiding).

[4] Analyse van organisatieproblemen, een toepassing van denken in systemen en processen  
Auteur: J. In 't Veld, 8<sup>e</sup> druk, Noordhoff Uitgevers B.V., 2002.

[5] Semantic Modelling & Linking Standard (SMLS), CEN TC442/WG4/TG3 (in ontwikkeling).

[6] ISO 15926 Part 11, *Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities — Part 11: Methodology for simplified industrial usage of reference data.*

[7] ISO/IEC 21320-1:2015, *Information Technology — Document Container File — Part 1: Core*  
<http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html>

[8] ISO 16354, *Guidelines for knowledge libraries and object libraries.*

[9] ISO 11354, *Advanced automation technologies and their applications — Framework for enterprise interoperability.*

[10] ISO 29481, *Building information models -- Information delivery manual -- Part 1: Methodology and format.*

[11] XML Schema Part 2: Datatypes, Second Edition, W3C Recommendation, 28 October 2004,  
<https://www.w3.org/TR/xmlschema-2/>.

[12] RDF 1.1 Concepts and Abstract Syntax, W3C Recommendation, 25 February 2014,  
<https://www.w3.org/TR/rdf11-concepts/>.

[13] SKOS Simple Knowledge Organization System Reference, W3C Recommendation, 18 August 2009,  
<https://www.w3.org/TR/skos-reference/>.

[14] RDF Schema 1.1, W3C Recommendation, 25 February 2014,

[15] <https://www.w3.org/TR/rdf-schema/>.

[16] OWL 2 Web Ontology Language, Document Overview (Second Edition), W3C Recommendation, 11 December 2012,  
<https://www.w3.org/TR/2012/REC-owl2-overview-20121211/>.

[17] <https://www.w3.org/TR/2012/REC-owl2-overview-20121211/>.

[18] Shapes Constraint Language (SHACL), W3C Recommendation, 20 July 2017,  
<https://www.w3.org/TR/shacl/>.

[19] SHACL Advanced Features, W3C Working Group Note, 08 June 2017,  
<https://www.w3.org/TR/shacl-aff/>.

[20] BP4mc2, Best Practices for meaningful connected computing, <https://bp4mc2.org/20181107/>,  
Specifiek profiel: Nederlands profiel voor gegevenscatalogi, Editor's Draft 11 September 2019,  
<https://bp4mc2.org/profiles/>.

[21] Metamodel Informatiemodellering (MIM), Geonovum Standaard, Werkversie 09 december 2019,  
<https://geonovum.github.io/MIM-Werkomgeving/>.

**Met opmerkingen [wb24]:** Er moeten in de tekst nog veel verwijzingen worden toegevoegd, bijv. [2] (zie onderstaande commentaren).

**Met opmerkingen [wb25]:** Van 6.2.6

NB: Deze reeks bevat veel ingetrokken delen.

**Met opmerkingen [wb26]:** Van 7.12.1.3

**Met opmerkingen [wb27]:** Ondergebracht in bibliografie.

**Met opmerkingen [wb28]:** Van 7.7.2

Bedoelen jullie ISO/IEC 10646,  
Information technology - Universal Coded Character Set (UCS)  
?

**Met opmerkingen [wb29]:** Afkomstig van 7.7.2

**Met opmerkingen [wb30]:** Naar h 2. Met de juiste referentie.  
Zie eerder commentaar over deze norm.

**Met opmerkingen [BH(31R30):** gaat tom doen met paul

**Met opmerkingen [wb32]:** Deze verwijzing ([7] of de norm) komt niet voor in de tekst.

**Met opmerkingen [BH(33R32):** is ook heel generiek..ik stel voor weghalen!

**Met opmerkingen [wb34]:** Hier schrappen, staat al in h 2. [8] komt ook niet voor als verwijzing in de tekst.

**Met opmerkingen [wb35]:** NEN-ISO 11354-1 (zie h 2)

**Met opmerkingen [wb36]:** Verwijzing staat niet in de tekst (naam van de norm ook niet).

**Met opmerkingen [wb37]:** Staat in h 2

**Met opmerkingen [wb38]:** Verwijzing staat niet in de tekst.

**Met opmerkingen [wb39]:** Staat in h 2

**Met opmerkingen [wb40]:** Staat in h 2.

**Met opmerkingen [wb41]:** Staat in h 2

**Met opmerkingen [wb42]:** Verwijzing staat niet in de tekst.

**Met opmerkingen [wb43]:** Verwijzing staat niet in de tekst.

**Met opmerkingen [wb44]:** Staan in h 2

**Met opmerkingen [wb45]:** Verwijzing staat niet in de tekst.

**Met opmerkingen [BH(46R45):** opgenomen bij annotaties

- [22] *OPM, An Ontology for describing properties that evolve over time*, geschreven door Mads Holten Rasmussen (DTU), Maxime Lefrançois (Uni Lyon), Mathias Bonduel (KU Leuven), Christian Anker Hviid (DTU) & Jan Karlshøj (DTU).
- [23] QUDT, <http://github.com/qudt/qudt-public-repo>.
- [24] RDF\*, <http://blog.liu.se/olafhartig/2019/01/10/position-statement-rdf-star-and-sparql-star/>

**Met opmerkingen [wb47]:** Titel en auteurs opnemen in bibliografie.  
Hier alleen de titel vermelden.

**Met opmerkingen [BH(48R47):** check: ref opgenomen

**Met opmerkingen [BH(49R47):** heb nu [22] opgemene bij eerste gebruik...ok?