

API Standaarden beheermodel

Logius Handreiking

Werkversie 09 maart 2026

**Deze versie:**

<https://logius-standaarden.github.io/API-Standaarden-Beheermodel/>

Laatst gepubliceerde versie:

<https://gitdocumentatie.logius.nl/publicatie/api/beheermodel/>

Laatste werkversie:

<https://logius-standaarden.github.io/API-Standaarden-Beheermodel/>

Vorige versie:

<https://gitdocumentatie.logius.nl/publicatie/api/beheermodel/1.0/>

Redacteurs:

[Logius Afdeling standaarden](#)

Alexander Green ([Logius](#))

Edwin Wisse ([Logius](#))

Peter Haasnoot ([Logius](#))

Auteur:

[Martin van der Plas](#) ([Logius](#))

Doe mee:

[GitHub Logius-standaarden/API-Standaarden-Beheermodel](#)

[Dien een melding in](#)

[Revisiehistorie](#)

[Pull requests](#)

Dit document is ook beschikbaar in dit niet-normatieve formaat: [PDF](#)



Dit document valt onder de volgende licentie:

[Creative Commons Attribution 4.0 International Public License](#)

Status van dit document

Dit is een werkversie die op elk moment kan worden gewijzigd, verwijderd of vervangen door andere documenten. Het is geen door het TO goedgekeurde consultatieversie.

§ Conformiteit

Naast onderdelen die als niet normatief gemarkeerd zijn, zijn ook alle diagrammen, voorbeelden, en noten in dit document niet normatief. Verder is alles in dit document normatief.

Samenvatting

Dit document beschrijft het beheermodel van de API Standaarden die Logius beheert. Dit model is gebaseerd op BOMOS.

§ 1. Inleiding

§ 1.1 Leeswijzer

Dit document beschrijft hoe Logius, afdeling Standaarden (hierna: Logius) de API Standaarden beheert en hoe de bijbehorende governance is ingericht.

§ 1.1.1 Bijlagen

Practische aspecten van het beheer, zoals de gebruikte applicaties en webservices zijn opgenomen in bijlagen van dit document. De bijlagen zijn niet specifiek voor een standaard maar zijn relevant voor alle standaarden onder beheer bij Logius.

§ 1.2 De API Standaarden

Het begrip API Standaarden is een verzameling van meerdere standaarden. We verstaan hieronder op dit moment de volgende standaarden:

- NLGov REST API Design Rules (In dit document wordt verder ADR gebruikt als afkorting).
- NLGov API Strategie modules (In dit document wordt verder Modules gebruikt als afkorting).

- NLGov Assurance profile for OAuth 2.0 (In dit document wordt verder OAuth-NL gebruikt als afkorting).
- OpenID NLGov (In dit document wordt verder OIDC-NL gebruikt als afkorting).
- NLGov profile for CloudEvents (In dit document wordt verder CloudEvents gebruikt als afkorting).

De API standaarden omvatten een sets van normatieve afspraken voor het structureren, beveiligen, autoriseren, identificeren en documenteren. De standaarden hebben tot doel om betere, uniforme en ontwikkelaar vriendelijke API's te ontwikkelen die makkelijk te implementeren zijn. De sets van afspraken bestaan uit breed toepasbare en ondubbelzinnige richtlijnen. Deze helpen organisaties die nieuwe API's ontwikkelen voor Nederlandse overheden (Rijk, provincies, gemeenten en waterschappen) en instellingen uit de (semi-) publieke sector. In onderstaande tabel zijn alle actuele links opgenomen naar alle bronnen van de standaarden:

Formele standaard	Gepubliceerde versie	Werk versie	Repository
NLGov API Design Rules (ADR)	2.1.0	2.2.0	API-Design-Rules
NLGov OAuth 2.0 profile (OAuth)	1.1.0	1.2.0	OAuth-NL-profiel
NLGov OpenID Connect profile (OIDC)	1.0.1	1.0.1	OIDC-NLGOV
Kennisplatform API's modulen	Stabiele modules, 21 december 2023	nvt	Kennisplatform APIs
CloudEvents	1.0	1.1	NL-GOV-profile-for-CloudEvents
Logboek dataverwerkingen	Nog niet vastgesteld	1.0.0	logboek-dataverwerkingen

§ 1.2.1 Nut

De overheid ontsluit gegevens en applicaties steeds vaker met standaarden. Voorbeelden hiervan zijn te zien op de website developer.overheid.nl, in Common Ground, Haal Centraal en het Digitaal Stelsel Omgevingswet.

Representational state transfer (REST) is een ontwerpprincipe dat wereldwijd veel gebruikt wordt voor het bouwen van programmeerinterfaces over het web (API's). REST is geen standaard maar een ontwerpprincipe, en laat nog veel vrijheid in het structureren van API's.

Deze standaarden maken het voor ontwikkelaars gemakkelijker om betrouwbare applicaties te ontwikkelen met API's van de overheid.

§ 1.2.2 Werking

Een application programming interface (API) is een gestructureerd en gedocumenteerd koppelvlak voor communicatie tussen applicaties. Zo lang er computers zijn, bestaan er API's en worden er verschillende API technologieën gebruikt. Sinds 2004 heeft Representational state transfer (REST) zich ontwikkeld tot een bepalend principe voor het realiseren van API's. Zogenaamde 'REST-API's' doen voor applicaties wat websites voor mensen doen. Websites presenteren informatie aan mensen, REST-API's maken applicaties en gegevens over het Internet beschikbaar voor andere applicaties. De technologie achter websites en REST-API's heeft daarom veel gemeen.

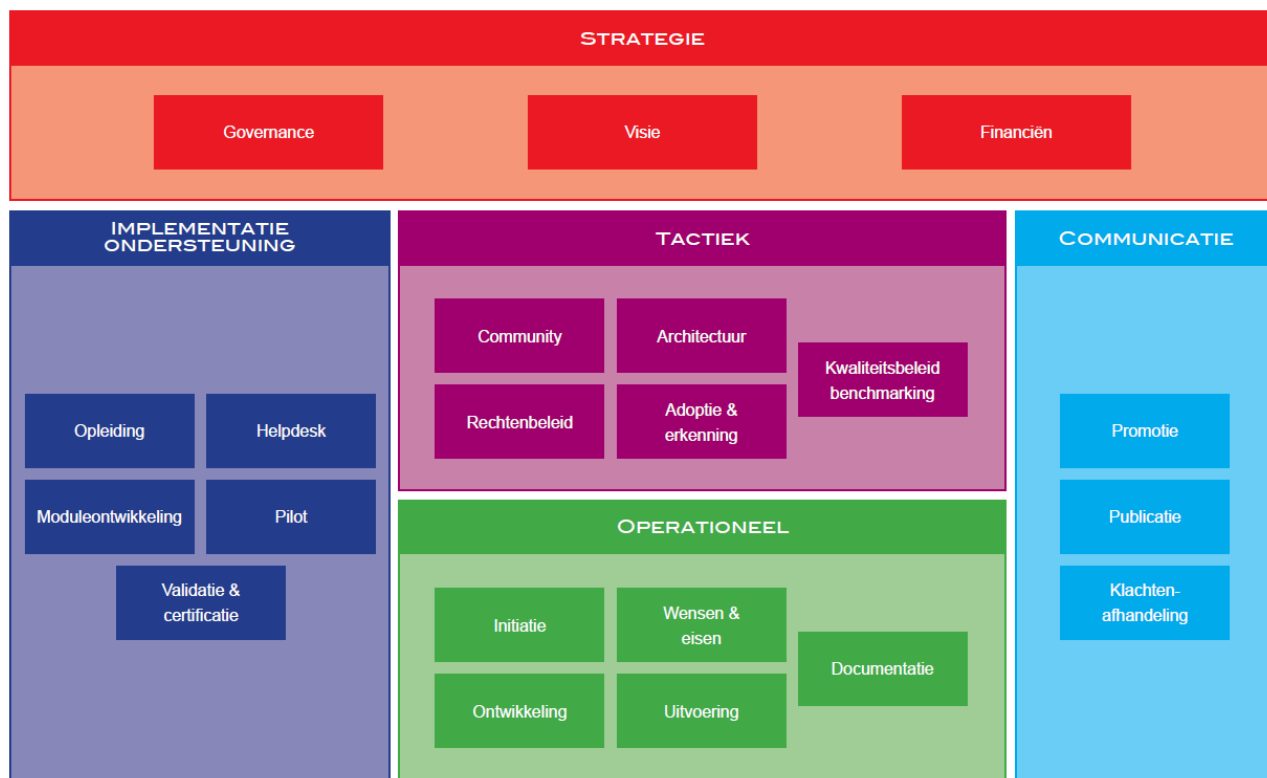
De overheid gebruikt REST-API's voor koppelingen met andere overheden, bedrijven en indirect ook met burgers, bijvoorbeeld via mobiele apps en webapps die aangeboden worden door bedrijven of overheden zelf. Ontwikkelaars kunnen deze REST-API's bevragen vanuit de gangbare programmeertalen en frameworks zoals Python, Java, Microsoft C#, PHP.

§ 1.2.3 Status

De statussen van de verschillende API standaarden zijn in de standaarden zelf vastgelegd. Dit beheermodel is initieel goedgekeurd in het Technisch Overleg, waarna het beheermodel is ingediend bij Mido en Forum Standaardisatie in combinatie met het indienen van de ADR als standaard. Met het indienen van versie 2.0 van de ADR standaard zijn tevens de beheermodellen van ADR en OAuth (die hadden een apart beheermodel) samengevoegd met dit beheermodel. Ook deze samenvoeging is goedgekeurd en de kopieën zijn daarom nu gearchiveerd.

§ 1.3 BOMOS

Het activiteitendiagram toont welke lagen het model onderscheidt en welke activiteiten daarbinnen onderscheiden worden. De lagen en de ondersteunende activiteiten worden elk in een hoofdstuk besproken.



Figuur 1 BOMOS activiteitendiagram

Voor meer details of BOMOS verwijzen we naar de documentatie: [BOMOS, het fundament](#) en [BOMOS, de verdieping](#)

§ 2. Strategie

De strategische activiteiten van BOMOS bestaan uit de onderdelen Visie, Governance en Financiering. Deze onderdelen en hun toepassing op het beheer van de API Standaarden worden hieronder beschreven.

§ 2.1 Visie

Met de API Standaarden wil de Nederlandse overheid interoperabiliteit bevorderen. Dit komt erop neer dat overheden dezelfde standaard in vergelijkbare situaties toepassen. Dit maakt uiteindelijk dat componenten en systemen onderling effectief gegevens uit kunnen wisselen. Zowel horizontaal in één voorziening binnen één situatie als verticaal tussen voorzieningen in verschillende situaties en tussen organisaties. Deze doelstelling wordt onderschreven door een breed scala aan partijen die deelnemen aan het API Kennisplatform, waar de ontwikkeling van de standaard zijn oorsprong heeft.

De API Standaarden worden bestendigd door het Forum Standaardisatie en het Overheidsbrede Beleidsoverleg Digitale Overheid (OBDO), die meerdere API Standaarden al hebben opgenomen op de zogenaamde ‘pas toe of leg uit’-lijst met standaarden die interoperabiliteit bevorderen [zie ook de basisinformatie van het Forum Standaardisatie](#).

De toetsingsprocedure voor opname van een standaard op de *pas toe of leg uit* -lijst bestaat uit de volgende stappen:

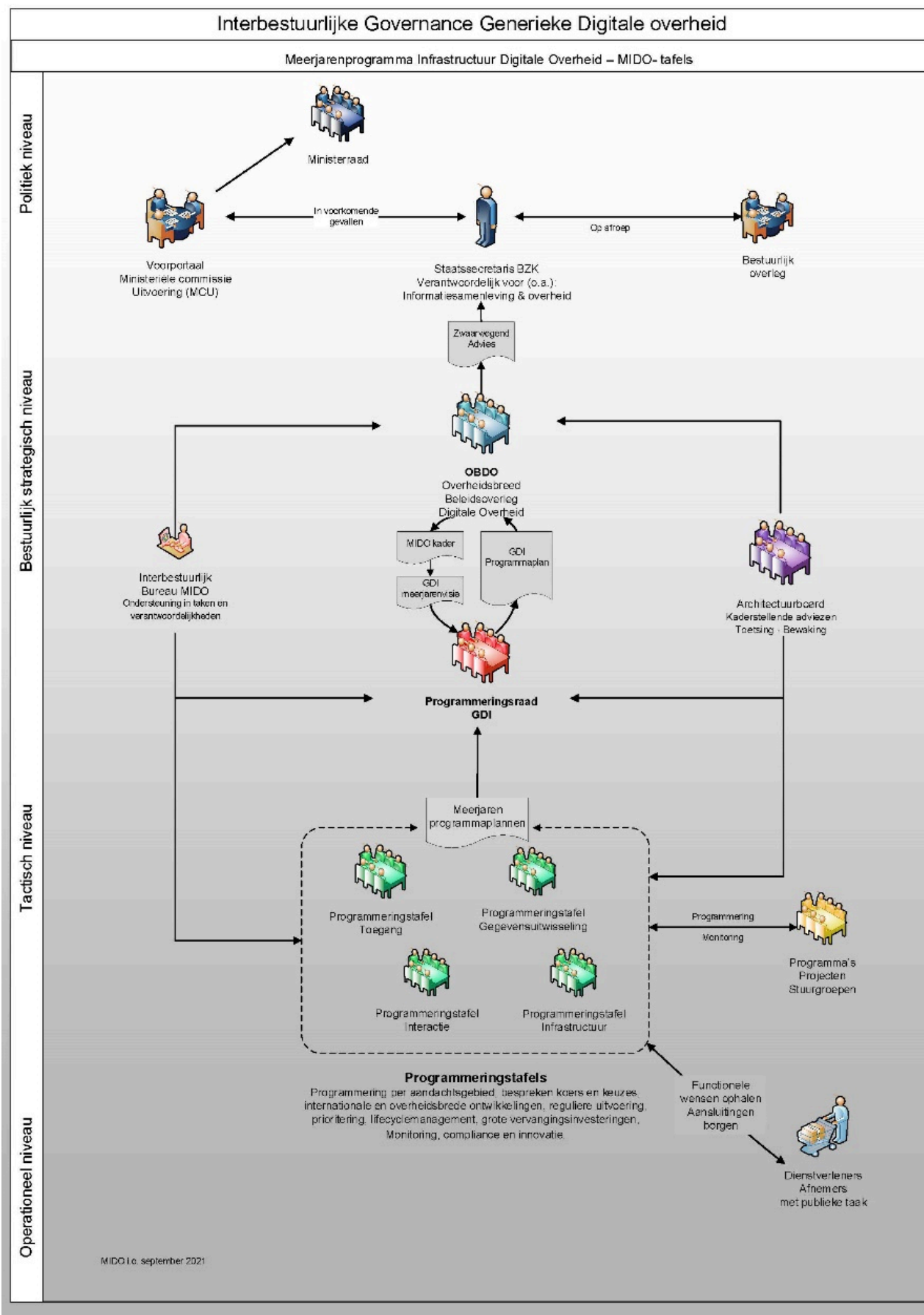
1. Aanmelding
2. Intake
3. Expertonderzoek
4. Openbare consultatie
5. Advisering door het Forum Standaardisatie
6. Vaststelling door het Overheidsbrede Beleidsoverleg Digitale Overheid

Deze criteria staan op: [Toetsingsprocedure en criteria voor lijsten met open standaarden \(forumstandaardisatie.nl\)](#)

§ 2.2 Governance

Bij het beheer van een open standaard hoort een open governance en een open procedure voor belanghebbenden om te kunnen participeren in het beheer. Logius neemt hierin de rol van onafhankelijke, duurzame beheerpartij en facilitator. Logius gaat uit van de governance van de Generieke Digitale Infrastructuur (GDI).

De GDI geeft richting aan het Meerjarenprogramma Infrastructuur Digitale Overheid (MIDO). Voor MIDO is een governance opgesteld waarin de stakeholders van Logius richting geven aan de ontwikkelingen bij Logius. Standaardenbeheer sluit aan op deze governance.



Figuur 2 MIDO governance

De MIDO governance kent vier *programmingstafels* op de thema's *Gegevensuitwisseling*, *Infrastructuur*, *Interactie* en *Toegang*. Op de tafels wordt de ontwikkeling en prioritering van de door Logius beheerde stelsels, standaarden en diensten besproken met de stakeholders.

§ 2.2.1 Governancestructuur

Bij het beheer van een open standaard hoort een open governance en een open procedure voor belanghebbenden om te kunnen participeren in het beheer. Logius afdeling standaarden neemt hierin de rol van onafhankelijke, duurzame beheerpartij en facilitator. Bij het beheer van de API Standaarden worden verschillende gremia onderscheiden die gezamenlijk invulling geven aan de governance op de standaard:

§ 2.2.1.1 Het Kennisplatform API's (community)

Dit is het meest operationele gremium waarin iedere belangstellende/belanghebbende vragen kan stellen over de API Standaarden en suggesties kan doen voor de doorontwikkeling van de standaard. Dergelijke vragen en suggesties worden door Logius Standaardenbeheer verzameld en voorgelegd aan het Technisch Overleg en als issue geregistreerd.

Omdat iedere belangstellende vragen of voorstellen tot wijziging in kan dienen is het niet nodig lid te worden van de community om een bijdrage te leveren. Iedereen die bijdraagt aan de standaard is daarmee lid van de community.

§ 2.2.1.2 Technisch Overleg

Het Technisch Overleg (TO) is een periodieke bijeenkomst waarbij de vragen en doorontwikkelwensen m.b.t. de API Standaarden worden doorgenomen, geprioriteerd en worden uitgewerkt. Daarnaast wordt door de leden de releaseplanning en de roadmap opgesteld. Deelname aan het Technisch Overleg is vrij voor eenieder die een belang heeft bij de standaard (overheid, wetenschap en markt).

De agenda en stukken van het Technisch overleg zijn openbaar.

§ 2.2.1.3 Tactisch overleg: de programmeringstafel

Dit overleg is verantwoordelijk voor het vaststellen van de doorontwikkel-roadmap, het vaststellen van major/minor releases van de standaard en dient als het voorportaal van het strategisch/besluitvormende gremium: de Programmeringsraad GDI.

Binnen de MIDO structuur hebben de Programmeringstafels de rol van tactisch overleg. Afhankelijk van het thema is die de Programmeringstafel Gegevensuitwisseling, Infrastructuur of Interactie.

§ 2.2.1.4 Het Strategisch overleg: De Programmeringsraad GDI

In de MIDO structuur heeft de Programmeringsraad GDI (PGDI) een rol in het strategisch beheer van standaarden. De programmeringsraad is gemandateerd (door het [OBDO](#)) om besluiten te nemen over wijzigingen op de standaard.

Het strategisch overleg keurt voorstellen tot wijziging goed op basis van adviezen van het technisch overleg, het tactisch overleg en het advies van de beheerorganisatie. Daarnaast keurt het strategisch overleg de door de stakeholders voorgestelde richting goed die aan de beheerorganisatie voorgelegd wordt. Bijvoorbeeld een voorstel tot ingrijpende wijziging zoals het overgaan naar een nieuwe (onderliggende) standaard kan in het strategisch overleg goedgekeurd worden.

De beheerorganisatie werkt goedgekeurde voorstellen uit en neemt deze op in een vast te stellen nieuwe versie.

§ 2.2.1.5 Overheidsbreed Beleidsoverleg Digitale Overheid (OBDO)

Het Overheidsbreed Beleidsoverleg Digitale Overheid (OBDO) is het overkoepelend overleg voor de MIDO overleggen. Formeel vindt besluitvorming plaats op het niveau van het OBDO. Voor GDI standaarden mandateert het OBDO de programmeringsraad tot het nemen van besluiten over wijzigingen op de standaarden.

Het OBDO wordt geïnformeerd over wijzigingen op de standaarden.

In tabelvorm:

Gremium	Accent	Rol participant	Ondersteuning door beheerder (Logius)
Community (omvang beperkt)	Inhoud & kennis - delen	1. Volgen van ontwikkelingen. 2. Leveren van input voor de doorontwikkeling van de standaard.	1. Informatie m.b.t. specificaties en beheer open delen met community. 2. Deelnemen aan stuurgroep en werkgroepen
Technisch Overleg (Operationeel, 4x per jaar)	Inhoud - afstemmen	1. Inhoudelijk ontwikkelen van standaard onderdelen en bijbehorende documentatie. 2. Voorbereiden van de release- planning. 3. Prioriteiten stellen voor de ontwikkeling, roadmap van nieuwe releases van de standaarden. 4. Goedkeuring van aanpassingen op de standaard. 5. Advies aan programmeringstafel en - raad over wijzigingsvoorstellen.	1. Analyseren, ontwerpen en uitwerken van specificaties. 2. Volgen en beïnvloeden van aanpalende standaarden. 3. Organiseren bijeenkomsten. 4. Opstellen en verspreiden notulen. 5. Beschikbaar stellen specificaties.
Programmeringstafel	Adviserend	1. Goedkeuren roadmap van de standaard. 2. Goedkeuren major/minor releases van de standaard.	1. Analyseren, ontwerpen en uitwerken van beleidszaken, (release)planning.
Programmeringsraad	Besluitvormend	1. Goedkeuren van grote wijzigingen: Introductie nieuwe API standaarden en uitfasering bestaande API standaarden. 2. Goedkeuren beheermodel van de standaard.	1. Advisering en inbreng via secretariaat MIDO. 2. Publiceren standaarden en andere Standaard-informatie

Gremium	Accent	Rol participant	Ondersteuning door beheerder (Logius)
		3. Goedkeuren externe publicaties over het standaardenbeleid en releases. 4. Goedkeuren major/minor releases van de standaard.	
OBDO	Besluitvormend	Het OBDO wordt geïnformeerd over wijzigingen op de standaard.	Toelichten wijzigingen in een release

§ 2.2.1.6 Architectuurraad

De Architectuurraad GDI van de MIDO governance maakt geen deel uit van het beheerproces. Wel kan de beheerder advies vragen over een wijzigingsvoorstel. Dit kan gevraagd worden op eigen initiatief of op initiatief van het Technisch Overleg.

§ 2.2.2 Besluitvorming

Besluitvorming over wijzigingsvoorstellen kan plaatsvinden op verschillende niveaus.

In alle overlegsgremia vindt oordeelvorming plaats op basis van consensus. Mocht consensus niet mogelijk zijn, dan gaat het vraagstuk met een weergave van de verschillende standpunten door naar het eerstvolgend-hoger gelegen-gremium. Indien in het hoogste gremium (het OBDO) geen consensus bereikt kan worden heeft de voorzitter van het OBDO (ministerie van BZK) de beslissende stem.

Voor wijzigingen met zeer kleine impact (tekst correcties) wordt de beheerorganisatie gemandateerd. De beheerorganisatie mag deze wijzigingen zelf doorvoeren zonder formele beslissing door het besluitvormend overleg. In de [versienummering](#) worden deze zeer kleine wijzigingen aangeduid als *patch releases*. Voor andere wijzigingen is een besluit van het PGDI nodig (op basis van advies van de Programmeringstafel en de beheerorganisatie). Het OBDO wordt geïnformeerd over wijzigingen op de standaard.

§ 2.2.3 Deelname

Uitbreidingen en aanpassingen in de API Standaarden komen tot stand door participatie van de verschillende belanghebbenden. Belanghebbenden kunnen op vijf manieren participeren aan het wijzigings- en besluitvormingsproces:

1. Als lid van de Community. Er is geen formeel lidmaatschap nodig om een issue/wijziging in te dienen. Iedereen die een issue indient is daarmee lid van de community.
2. Als lid van de Technisch Overleg
 1. Leden van het technisch overleg dienen een aantoonbaar belang te hebben bij de standaard.
 2. De omvang en samenstelling moet een goede vertegenwoordiging bevatten van de verschillende belangen rond de standaard. We gaan uit van 1 deelnemer per organisatie.
 3. Het belang van de Nederlandse overheid dient voldoende geborgd te zijn in het technisch overleg.
3. Als lid van de Programmeringstafel Gegevensuitwisseling
 1. Stakeholders van de Logius Gegevensuitwisselingsdiensten worden uitgenodigd.
4. Als lid van de Programmeringsraad GDI.
5. Als lid van het OBDO.

Personen/partijen die willen deelnemen aan het Technisch Overleg kunnen contact opnemen met Logius waarin zij aangeven wat hun belang is bij de standaard. Met inachtneming van bovenstaande punten, beoordeelt Logius de aanvraag.

§ 2.3 Financiering

Het beheer van de API Standaarden werd initieel gefinancierd door min. BZK voor een periode van drie jaar (2020-2023) om gebruikers het vertrouwen te geven dat er geen desinvesteringen worden gedaan bij het implementeren van de standaarden. Na die drie jaar werd de financiering verlengd, aangezien het nut en de behoefte aan de standaarden was aangetoond.

§ 3. Tactiek

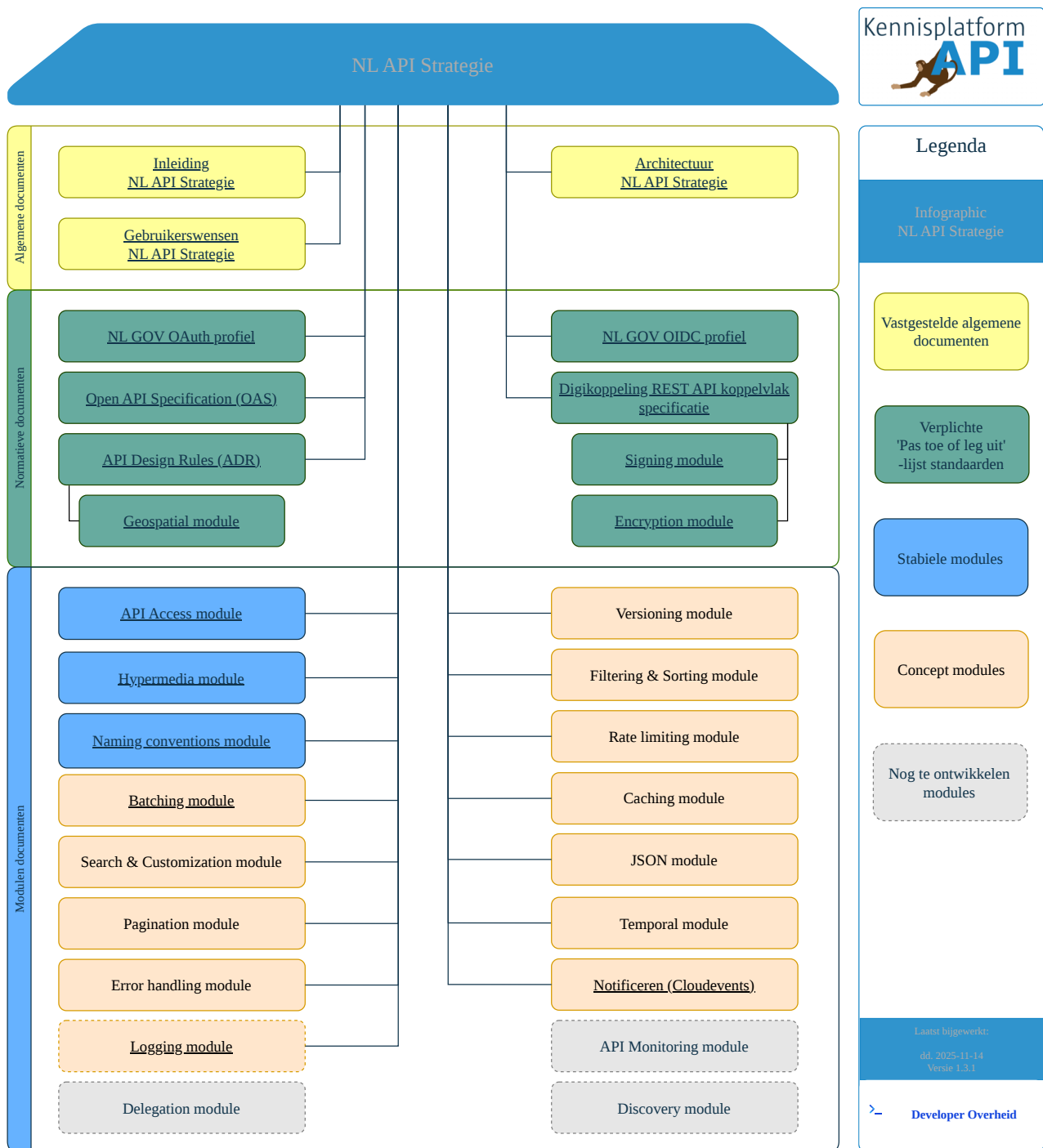
§ 3.1 Community

De community bestaat uit eenieder die belanghebbende of belangstellende is m.b.t. de standaard. Deelname aan de community kent geen drempels of restricties. Leden van de community kunnen alle informatie m.b.t. de standaard en het beheer daarvan inzien via de website en via verschillende kanalen issues of RFC's melden. Daarnaast kunnen community leden reageren op openbare consultaties en onder bepaalde voorwaarden deelnemen aan de Technische Architectuur Groep (zie [2.2.3 Deelname](#)).

§ 3.2 Architectuur

De API Standaarden een op zichzelf staande standaarden en geen onderdeel van een bovenliggende standaard. Wel wordt er onderling tussen de standaarden en zijn er verwijzingen naar verschillende andere (internationale) standaarden.

Een overzicht van alle API Standaarden is gepubliceerd in deze infographic:



Figuur 3 NL API Strategie Infographic

§ 3.2.1 Internationale, Europese en nationale standaardisatiegemeenschap

De ADR-Standaard volgt de ontwikkeling van internationale standaarden (zoals bijvoorbeeld de HTTP standaarden van het IETF) in het algemeen. Meer specifiek volgen de specialisten van Logius en de leden van de TAG de standaarden waarnaar wordt gerefereerd in de API Standaarden en bespreken deze ontwikkelingen ook in het Technisch Overleg. Indien relevant worden op basis van de internationale ontwikkelingen rfc's opgesteld om de API Standaarden aan te passen,

verbeteren of actualiseren. Onderstaand is het overzicht overgenomen van de standaarden waaraan wordt gerefereerd in de ADR:

1. **[OPENAPIS]**

OpenAPI Specification. Darrell Miller; Jeremy Whitlock; Marsh Gardiner; Mike Ralphson; Ron Ratovsky; Uri Sarid; Tony Tam; Jason Harmon. OpenAPI Initiative. URL: <https://www.openapis.org/>

2. **[rfc3986]**

Uniform Resource Identifier (URI): Generic Syntax. T. Berners-Lee; R. Fielding; L. Masinter. IETF. January 2005. Internet Standard. URL: <https://datatracker.ietf.org/doc/html/rfc3986>

3. **[rfc5789]**

PATCH Method for HTTP. L. Dusseault; J. Snell. IETF. March 2010. Proposed Standard. URL: <https://httpwg.org/specs/rfc5789.html>

4. **[rfc7231]**

Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. R. Fielding, Ed.; J. Reschke, Ed.. IETF. June 2014. Proposed Standard. URL: <https://httpwg.org/specs/rfc7231.html>

5. **[SemVer]**

Semantic Versioning 2.0.0. T. Preston-Werner. June 2013. URL: <https://semver.org>

In de OAuth-NL, OIDC-NL en Modules wordt naar nog veel meer standaarden verwezen. Alle verwijzingen zijn online gepubliceerd door de IETF, W3C, OpenID Foundation, CNCF en andere.

§ 3.2.2 Samenwerking met andere beheerorganisaties

§ 3.2.2.1 Kennisplatform API's

Kennisplatform API's is een initiatief van Geonovum, Bureau Forum Standaardisatie, Kamer van Koophandel, VNG Realisatie en Logius. Het doel van het Kennisplatform is om de kennis over het toepassen van API's uit te wisselen en de aanpak bij verschillende organisaties op elkaar af te stemmen en waar nodig te standaardiseren. In het kennisplatform wordt gezamenlijk gekeken naar strategische en tactische vraagstukken rond het ontwikkelen van API's door de overheid en gebruik van deze API's buiten en binnen de overheid. Dit vanuit de gedachte dat we in een digitale samenleving eenvoudig met elkaar moeten kunnen samenwerken.

De API Standaarden komen voort uit de Nederlandse API Strategie die beheerd wordt door het Kennisplatform API's en zijn veelal door het kennisplatform ontwikkeld. Op het moment dat er in het kennisplatform consensus was over de kwaliteit van de standaard en de wenselijkheid deze via het 'pas toe of leg uit' -principe normatief te laten verklaren is de standaard voorgedragen aan Forum Standaardisatie voor het verkrijgen van de voor overheden verplichte 'pas toe of leg uit' status en heeft Logius het beheer van dit normatieve deel op zich genomen.

Het kennisplatform API's blijft via haar werkgroepen actief met API Standaarden, maar richt zich primair op de ontwikkeling van nieuwe modules. Deze modules zijn bovendien (nog) niet normatief van aard. Logius heeft bij het beheer van de API Standaarden nauw contact met het kennisplatform om zo te borgen dat wensen en issues m.b.t. de API Standaarden bij beide partijen helder zijn en hier gezamenlijk de beste aanpak voor gekozen kan worden. (Zie ook [4. Operationeel](#)).

§ 3.2.3 Nederlandse Overheid Referentie Architectuur (NORA)

De API Standaarden volgen de principes van de Nederlandse Overheid Referentie Architectuur. Zie voor meer informatie: https://www.noraonline.nl/wiki/NORA_online

In de NORA is sinds 2017 het Thema API's opgenomen en beschreven. De NORA beschrijft met name wat een API is en waarom API's belangrijk zijn. Ook zijn er op de site aanbevelingen voor API's in de Enterprise Architectuur en de toepassing van API's in het ontwerp van een dienst.

§ 3.3 Rechtenbeleid

Dit werk is gelicenseerd onder een Creative Commons Naamsvermelding 4.0 Unported licentie.



[Figuur 4](#) Creative Commons Naamsvermelding 4.0 Unported licentie

Meer informatie over de precieze voorwaarden van deze licentie vindt u op de website van Creative Commons <http://creativecommons.nl/> en specifiek voor deze licentie een samenvatting onder <http://creativecommons.org/licenses/by/4.0/deed.nl> en de volledige licentietekst onder <http://creativecommons.org/licenses/by/4.0/legalcode>.

Dit werk en de specificaties van de API Standaarden worden royalty free ter beschikking gesteld. Organisaties en personen die bijdragen aan de API Standaarden dienen hun bijdragen vrij te geven zodanig dat hieraan voldaan kan worden. Door bij te dragen aan de API Standaarden verklaren zij hiermee in te stemmen.

Uitgesloten van alle bovenstaande zijn rechten verbonden aan de standaarden, profielen en andere onderdelen waar de API Standaarden gebruik van maakt. Hierop zijn de rechten van de betreffende standaarden, profielen en andere onderdelen zelf van toepassing.

§ 3.4 Kwaliteitsbeleid en benchmarking

Zoals gezegd wordt het beheer van de API Standaarden volledig open ingevuld (zie ook [1.3 BOMOS](#) en [2.2 Governance](#)) Dit borgt dat zoveel mogelijk belangstellenden en belanghebbenden betrokken zijn bij wijzigingen en besluitvorming over wijzigingen.

§ 3.5 Adoptie en erkenning

De meerderheid van de API Standaarden hebben de 'pas toe of leg uit' -status van Forum Standaardisatie. Dit betekent kort gezegd dat Nederlandse overheidspartijen en partijen uit de (semi) publieke sector deze standaarden dienen toe te passen op het moment dat zij hun informatie met behulp van API's willen ontsluiten. Zie hoofdstuk 1 voor meer informatie.

§ 4. Operationeel

Operationeel beheer omvat volgens BOMOS het tekstuele beheer van de documentatie, het verzamelen van eisen en wensen en de vertaling daarvan naar wijzigingsvoorstellen. Verder omvat het operationele proces de besluitvorming en het versie- of release-beheer.

Het operationele wijzigingsproces is ingericht op Github. De omgeving die we ook gebruiken voor het beheer en de publicatie van de documentatie. In dit hoofdstuk wordt het operationele wijzigingsproces op hoofdlijnen beschreven. Voor details van de implementatie verwijzen we naar [B. Gebruik GitHub in het beheerproces](#)

§ 4.1 Initiatie

Toevoegingen aan de API Standaarden zoals het toevoegen van een nieuwe module worden behandeld als introductie van een nieuwe standaard.

1. Uitbreidingen en aanpassingen in de standaarden in beheer bij Logius komen tot stand door participatie van de verschillende belanghebbenden.
2. Belanghebbenden kunnen op verschillende manieren participeren.
 1. op persoonlijke titel (het proces is volledig open)
 2. als lid van de Community
 3. als lid van één van de overleggen: het Technisch Overleg, de Programmeringstafel Gegevensuitwisseling of het OBDO.

§ 4.2 Wensen en Eisen

Wensen en eisen zijn aanpassingen op de bestaande API standaarden. Wijzigingsvoorstellen kunnen binnen komen via verschillende kanalen:

1. Rechtstreeks bij de beheerorganisatie, tijdens overleggen, via de website of mail
2. Bij de werkgroepoverleggen van de standaard en tijdens overleggen, via de website of mail

§ 4.3 Uitvoering en ontwikkeling (Wijzigingsproces)

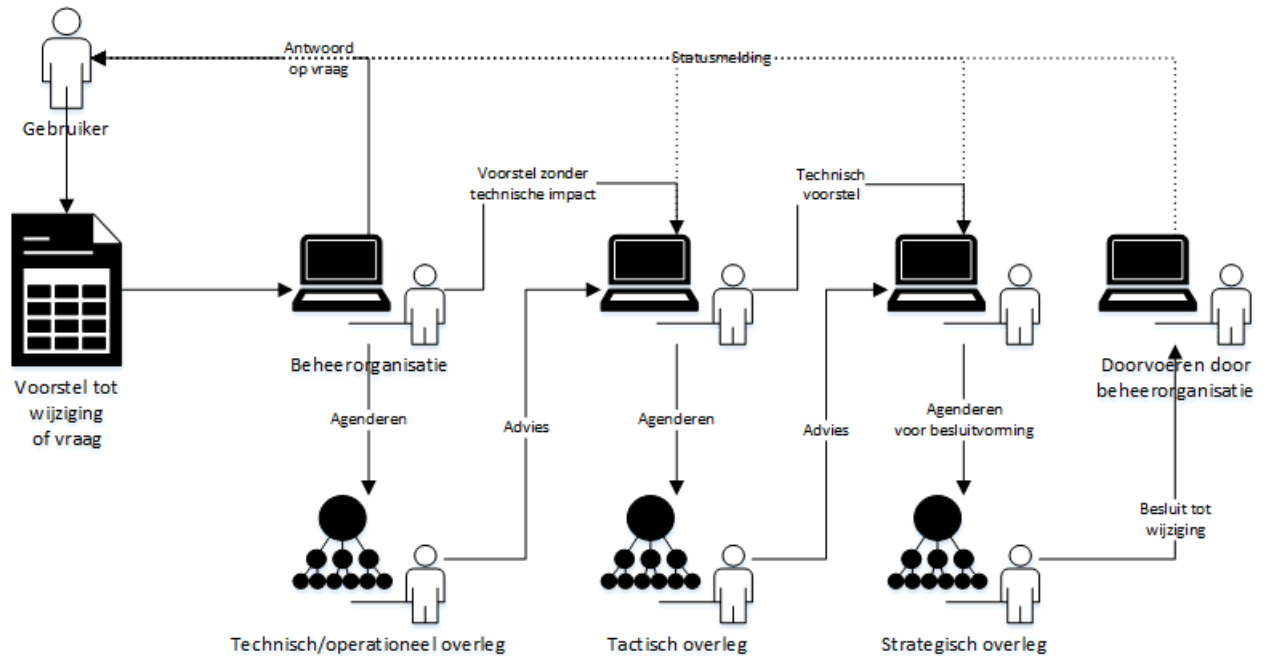
Afhankelijk van de impact van een wijziging kan deze aangemerkt worden als een *patch*. Een patch is een kleine (tekstuele) wijziging die geen impact heeft op implementaties.

Een *wijziging* is een aanpassing met impact op de werking of het proces van de API standaard. Waarbij nog een onderscheid gemaakt wordt tussen wijzigingen met kleine en met grote impact.

Patches en wijzigingen worden verzameld in een *release*. Een release is een nieuwe versie van de API standaard. Nieuwe releases worden regelmatig doorgevoerd en moeten worden goedgekeurd door het Technisch Overleg en, afhankelijk van de impact van een nieuwe release door een programmeringstafel. Een nieuwe release wordt bekrachtigd door het besluitvormend overleg.

§ 4.3.1 Wijzigingen

Onderstaand schema geeft een overzicht van de behandeling van een wijzigingsvoorstel in het beheerproces:



Figuur 5 Behandeling van een wijzigingsvoorstel in het beheerproces

Onderstaande stappen worden door de beheerorganisatie genomen om het proces te doorlopen:

1. Acceptatie van een wijzigingsvoorstel.
2. Labelen van een voorstel als groot/klein, aangeven van status.
3. Behandeling van een wijzigingsvoorstel.
4. Agendering voor een overleg.
5. Advisering vanuit overleggen.
6. Acceptatie van een wijzigingsvoorstel.
7. Doorvoeren van een wijzigingsvoorstel.
8. Publicatie van een wijziging in de komende release

§ 4.3.2 Patches

Een patch is een zeer kleine wijziging die geen impact heeft op de implementatie. Bijvoorbeeld tekstuele wijzigingen in de documentatie. De beheerorganisatie beoordeelt de impact van een wijziging en bepaalt daarmee of het een patch betreft (of een wijziging).

1. Beoordeling van een voorgestelde patch door de beheerorganisatie
2. Doorvoeren van de patch door de beheerorganisatie
3. Publicatie van een patch in een release

§ 4.3.3 Releases

De API Standaard zullen gezamenlijk en afzonderlijk onderhevig zijn aan beheer en onderhoud wat leidt tot nieuwe releases. Het vaststellen van nieuwe releases vindt plaats binnen het releaseplanningsproces. Het tactisch overleg is verantwoordelijk voor de juiste uitvoering. Hier komen alle belanghebbenden met verantwoordelijkheid voor de behoefte, effecten en impact op de bedrijfsvoering, informatievoorziening en ICT samen.

Voor nieuwe releases wordt uitgegaan van een aantal principes:

1. De standaard dient in principe zo stabiel te zijn dat nieuwe releases van de standaard bestaande implementaties van een oudere release niet tot migratie verplichten.
2. De standaard streeft ernaar om interoperabel (engels: backwards compatible) te zijn met voorgaande releases (interoperabele verandering). Bij wijzigingen waarin ook dit niet mogelijk is, vindt een expliciete afweging plaats van de geboden verbetering ten opzichte van het belang van bestaande implementatie (beperking impact).
3. Wijzigingsaanvragen kunnen door belanghebbenden worden ingediend bij de beheerder.
4. Het Technisch Overleg is verantwoordelijk voor de beoordeling van ingediende wijzigingsaanvragen, uitwerken ervan en de inhoudelijke (door)ontwikkeling van de te beheren API Standaarden.
5. De beheerder zorgt voor de voorbereiding van de releaseplanning.
6. Het tactisch overleg beoordeelt de releasevoorstellen en stelt het beleid en de roadmap van nieuwe releases vast in het releaseplanningsproces.
7. Bij het vaststellen van een nieuwe release kan het strategisch overleg uitspraken doen over het ondersteunen van oude releases.

8. Maximaal kunnen twee (opéénvolgende) releases van een API Standaard gelijktijdig de status „In Gebruik” hebben.
9. Op het moment dat het functionele toepassingsgebied van een API Standaard, waarvoor het pas-toe-of-leg-uit-regime geldt wijzigt, wordt dit voorgelegd aan Forum Standaardisatie en het OBDO zodat het regime kan worden bekrachtigd voor dit nieuwe toepassingsgebied.

§ 4.3.4 Impact van wijzigingen en versienummering

Afhankelijk van de impact van een wijziging of patch krijgt een release een nieuw versienummer. Het versienummerbeheer volgt principes voor semantische versienummering [en is beschreven in een bijlage](#)

De beheerorganisatie schat op basis van de wijziging of patch in welk nieuw versienummer noodzakelijk is.

§ 4.4 Status van de standaard

Afkorting	Status van de standaard	Beschrijving van de status
IO	In Ontwikkeling	Een nieuwe release van de standaard is "In Ontwikkeling" wanneer er met medeweten en medewerking van participanten aan gewerkt wordt en wanneer dit onderdeel of deze release nog niet voor de buitenwereld is gepubliceerd.
IG	In Gebruik	Als een nieuwe release van de standaard gereed is, en is bestendigd door Forum Standaardisatie, stelt het Technisch Overleg de status 'In Gebruik' vast. Door deze vaststelling worden gebruikers en ICT-leveranciers opgeroepen deze nieuwe release op te nemen in software en in gebruik te nemen.
EO	Einde Ondersteuning	De standaardversie met de status "Einde ondersteuning" wordt niet meer ondersteund door de beheerder. De kennis en informatie voor vragen en support is bij de beheerder niet langer beschikbaar.
TG	Teruggetrokken	De standaard krijgt de status "Teruggetrokken" indien een release van de standaard niet bruikbaar blijkt (bijv. vanwege implementatieproblemen).

§ 4.5 Documentatie

Alle documenten m.b.t. de API standaarden en het beheer van de standaarden worden openbaar en zonder drempels voor gebruik, gepubliceerd op logius.nl en onze [Github pagina's](#). Logius publiceert tenminste de volgende documenten:

- Dit API Standaarden beheermodel
- De vergaderstukken van het Technisch overleg en overige besluitvormende gremia.
- De specificaties van de standaard
- De voorlopige specificaties van de nieuwe versie van de standaard.

§ 5. Implementatieondersteuning

§ 5.1 Opleiding en advies

Logius biedt momenteel geen opleiding aan, maar borgt dat de informatie m.b.t. de standaard altijd, zonder drempels, toegankelijk is. Bovendien kunnen geïnteresseerden via verschillende kanalen contact opnemen met Logius in geval van vragen of opmerkingen. Zie hiervoor [5.2 Helpdesk](#). Aanvullend organiseert Kennisplatform API's regelmatig overleggen en seminars m.b.t. de Nederlandse API Strategie waar de ADR-standaard een onderdeel van is. Zie hiervoor www.apigov.nl.

§ 5.2 Helpdesk

Logius biedt ondersteuning en advies via verschillende kanalen:

- Online: als reactie op issue's in de GitHub van de standaard.
- Per mail: api@logius.nl
- Telefonisch: 0900 - 555 45 55
- Per post: Logius, Postbus 96810; 2509 JE Den Haag, (t.a.v. CvS).

§ 5.3 Validatie & Certificatie

Certificatie van API's is op dit moment niet mogelijk. Wel is het mogelijk API's te valideren en te testen met behulp van de door tools welke beschikbaar zijn op:

- [Developer.overheid.nl](https://developer.overheid.nl)

De resultaten van de tests zijn publiek beschikbaar op de site.

§ 6. Communicatie

§ 6.1 Promotie

De API Standaarden worden via verschillende kanalen gepromoot. Ten eerste via het Kennisplatform API's als onderdeel van de Nederlandse API-strategie. Naast communicatie op de website van het kennisplatform, organiseert het platform regelmatig vrij toegankelijke bijeenkomsten.

Daarnaast hebben meerdere API Standaarden de zogenaamde 'pas toe of leg uit' -status van Forum Standaardisatie. Dit betekent dat Forum Standaardisatie het gebruik van deze standaarden niet alleen actief promoot, maar in veel gevallen zelfs hard voorschrijft.

Tot slot is Logius promotor van de standaarden. Zowel intern voor de toepassing van de standaarden in Logius voorzieningen als extern, door andere partijen te informeren en adviseren over de mogelijkheden van de standaarden.

§ 6.2 Publicatie

Als een nieuwe versie van de API Standaarden de status "In Gebruik" heeft, worden verschillende zaken gepubliceerd.

Logius publiceert altijd de volledige specificaties van de standaarden op een deel van zijn website. Daarnaast wordt een persbericht uitgegeven, waarin de publicatie van nieuwe releases van de standaarden wordt aangekondigd.

Aanvullend publiceert Logius alle genoemde documentatie zoals genoemd bij [Documentatie](#).

§ 6.3 Klachtenafhandeling

Klachten over de opzet of de uitvoering van het beheerproces kunnen ingediend worden bij Logius. Dit kan in principe via alle beschikbare kanalen. De indiener van de klacht krijgt zo spoedig mogelijk en altijd terugkoppeling over de voortgang van en beslissing over zijn klacht. De volledige klachtenprocedure is terug te vinden in het generieke beheermodel van Logius, afdeling standaarden. (volgt)

§ A. Gebruik ReSpec

Voor publicatie van de standaarden die bij Logius in beheer zijn wordt gebruik gemaakt van ReSpec. ReSpec is een applicatie om technische documentatie te maken die publiceerbaar is op het net en gemakkelijk kan worden geïndexeerd door zoekmachines om de documentatie vindbaar te maken. Het is ontwikkeld ten behoeve van de documentatie van W3C standaarden. Door gebruik te maken van ReSpec publiceren we documentatie overeenkomstig een (de facto) W3C standaard.

ReSpec is een Javascript applicatie. Input voor ReSpec bestaat uit teksten in HTML of [Markdown](#), zie [[RFC7763](#)]. ReSpec combineert een serie input files tot één documentatiedocument in HTML met een duidelijke inhoudsopgave en kruisverwijzingen naar de verschillende secties en figuren.

ReSpec is ontwikkeld door een werkgroep van W3C en wordt actief doorontwikkeld. Meer informatie is gepubliceerd op respec.org.

§ A.1 Logius profiel

[Logius heeft een eigen profiel gemaakt op ReSpec](#) om Logius organisatiespecifieke zaken, zoals layout, te ondersteunen. Wijzigingen in de W3C versie worden regelmatig doorgevoerd in de Logius versie.

De Logius ReSpec versie is zo algemeen mogelijk gemaakt zodat deze door andere overheden in Nederland eenvoudig toegepast kan worden. In de Logius versie gebruiken we zoveel mogelijk input in Markdown formaat.

§ A.2 Literatuurverwijzingen

ReSpec maakt gebruik van de online [Specref](#) database van Literatuurverwijzingen. Deze database bevat referenties naar, onder andere, referenties voor de W3C documentatie.

Voor Nederlandse documenten die niet in Specref staan maken we gebruik van een standaard literatuurlijst die voor alle documenten gebruikt kan worden en die apart beheerd wordt. Het beheer is onder meer nodig om links naar online documentatie bij te houden.

§ A.3 Links

- [Markdown documentatie](#)
- [ReSpec documentatie](#)
- [Respec wiki op de repository van W3C](#)
- [Logius fork van Respec, gebruikt voor deze documentatie](#)

§ B. Gebruik GitHub in het beheerproces

§ B.1 Publicatie

[GitHub](#) biedt functionaliteit om documenten te publiceren vanuit een repository. Logius gebruikt deze functionaliteit om het met [ReSpec](#) gegenereerde document te publiceren als HTML-document en een PDF-document. Deze documenten worden automatisch gekopieerd naar een publicatiewebsite onder Logiusbeheer.

§ B.2 Wijzigingsvoorstellen

Het proces zoals beschreven onder [operationeel beheer, wensen en eisen](#) wordt voor de Logius standaarden geïmplementeerd door gebruik te maken van *GitHub issues*. Een *issue* kan binnen GitHub ingediend worden door iedere (GitHub)gebruiker en wordt bij ontwikkeling van code

gebruikt om functionele wensen of gevonden bugs in te dienen zodat deze door ontwikkelaars opgepakt kunnen worden. Een *issue* kan online besproken worden en uiteindelijk gesloten worden wanneer deze verwerkt is. Alle open en gesloten issue's blijven publiek inzichtelijk in de repository van de door Logius beheerde standaard.

§ B.2.1 Branches

Binnen het standaardenbeheer bij Logius maken we gebruik van verschillende branches. De *main* branch bevat de laatste formeel geaccepteerde versie van een document. De *develop* branch bevat een werkversie met daarin alle wijzigingen die in een volgende geaccepteerde versie opgenomen moeten worden.

Aanpassingen in de documentatie die voor een specifiek wijzigingsvoorstel gemaakt worden worden in eigen branch verwerkt. Deze branch wordt gesplitst vanaf de *develop* branch en wordt nadat het wijzigingsverzoek aangenomen is teruggebracht naar de *develop* branch.

Voorbeeld: een wijzigingsverzoek voor het aanpassen van de architectuurbeschrijving zal in een branche *nieuwe architectuur* worden verwerkt. Deze wordt gesplitst vanaf, en teruggebracht naar, de *develop* branch. Door wijzigingen in een eigen branch op te nemen zijn alle wijzigingen op de documentatie inzichtelijk per wijzigingsvoorstel.

De *develop* branch wordt dus niet gebruikt om wijzigingen op het document te maken maar dient als verzamelbranch voor de verschillende wijzigingen die in een volgende release moeten komen. *Patches* kunnen wel direct op de *develop* branch worden doorgevoerd.

§ B.2.2 Commits op develop en main

Deze sectie geldt niet voor commits die worden gemaakt tijdens het werk aan een pull request.

Commit messages zijn voor maintainers en derden belangrijk om een (historisch) beeld te krijgen van de evolutie van een standaard. Om het onderscheid duidelijk te maken tussen normatieve en niet-normatieve wijzigingen, moeten commits met enkel niet-normatieve wijzigingen starten met *Beheer:*. Probeer ook zoveel mogelijk wijzigingen te groeperen in commits die enkel normatieve wijzigingen toepassen en commits die enkel niet-normatieve wijzigingen bevatten. Als deze worden gegroepeerd in (grote) commits, dan wordt het voor de lezer (en voornamelijk implementeerders) lastiger om te weten wat er in code moet worden veranderd om aan de standaard te voldoen.

Voorbeeld van een commit met enkel niet-normatieve wijzigingen:

Voorbeeld van een commit met enkel normatieve wijzigingen:

Vereis dat alle URLs eindigen zonder een slash

Normatieve wijzigingen moeten geen prefix hebben en duidelijk uitleggen waar de wijziging over gaat, met in acht nemen van de generiek (Engelse) [guide voor het schrijven van bruikbare commit messages](#). In tegenstelling tot de gelinkte guide moeten commit messages in het Nederlands worden geschreven.

Om dit te vergemakkelijken wordt "Squash and merge" aangeraden. Dit stelt in staat om bij het mergen de commit message in de correct structuur te krijgen, zodat de commits tijdens de pull request flexibeler kunnen zijn.

In de commit message body wordt aangeraden om de volgende informatie toe te voegen:

- Achtergrond informatie wat de meerwaarde van de wijziging is
- Eventuele tips/tricks voor implementeerders hoe een wijziging kan worden geïntegreerd in bestaande implementaties
- Link naar het TO waarin de wijziging is geaccepteerd
- GitHub notatie van issues die hiermee opgelost zijn

§ B.2.3 Labels

Om GitHub issues te classificeren en te agenderen voor het juiste overleg maken we gebruik van een aantal standaard labels. We labelen binnenkomende issues als:

1. **Type** Alle soorten issues kunnen binnenkomen. Met Type sorteren we de issues in vragen, correcties en wijzigingen.
 1. Correctie
 2. Documentatie
 3. Vraag
 4. Wijziging
2. **Scope** Vooral relevant voor wijzigingsvoorstellen. Hiermee wordt aangegeven of het een kleine of grote wijziging betreft. Dit heeft betrekking op de impact van een wijziging en daarmee op de [versienummering](#).
 1. Klein

2. Groot

3. **Overleg** Het label Overleg heeft alleen betrekking op wijzigingsvoorstellen. Wanneer deze labels gebruikt worden wordt het voorstel geagendeerd voor het betreffende overleg.

1. TO-DK

2. TO-Auth

3. Gegevensuitwisseling

4. Toegang

5. Interactie

6. Infrastructuur

4. **Status**

1. In onderzoek

2. In bewerking

3. Uitwerking door derden

4. In review

5. Klaar voor review

6. Gereed

7. Afgewezen

§ B.3 Automatisering en scripts

GitHub ondersteunt automatisering van taken door scripts. Standaard is de publicatie via *github pages*. Binnen de Logius standaarden maken we gebruik van scripts om documenten te publiceren, links te checken en om een paar eenvoudige tests op digitoegankelijkheidseisen uit te voeren.

§ C. Versienummering

Deze bijlage beschrijft de versioneringsmethodiek ofwel de standaard manier om om te gaan met versie nummers van de standaard. De versioneringsmethodiek is gelijk voor alle 'gepubliceerde standaarden' die onder beheer zijn van Logius (afdeling standaarden) en is gebaseerd op SemVer. SemVer staat voor Semantisch Versioneren en we gebruiken versie 2.0.0 van de standaard zoals [gepubliceerd in de specificatie van Semantisch Versioneren \(SemVer\)](#).

Dat wil zeggen we kennen een bepaalde betekenis toe aan Major, Minor en Patch wijzigingen voor de standaarden zodanig dat de versienummers informatief zijn voor het type wijziging.

Aandachtspunt hierbij is dat semantische versionering voor standaarden anders werkt dan semantische versionering voor software.

De versienummers voor standaarden drukken uit of een (implementatie) van een oude versie van de standaard voldoet aan de regels van de nieuwe standaard (en dus compliant is aan de nieuwe versie) of niet. Het voordeel van deze manier van versioneren is dat het versienummer signaleert of een implementatie van een bepaalde versie van de standaard voldoet aan een andere (nieuwe) versie van de standaard of dat er sprake is van nieuwe / gewijzigde regels waar actie op moet worden ondernomen om compliant te blijven aan deze nieuwe versie.

De beschreven methodiek is van toepassing op de standaarden die Logius in beheer heeft. In de tekst worden Digikoppeling standaarden als voorbeeld aangehaald maar semantische versienummering is ook op de andere standaarden van toepassing.

§ C.1 Versioneringsmethodiek

Per document wordt met [documentnaam] X.Y.Z de versie aangegeven. Met X.Y.Z wordt gerefereerd aan major (X) en minor (Y) releases en (Z) patches, dit wordt hieronder toegelicht.

- PATCH wordt verhoogd bij correcties.
- MINOR wordt verhoogd bij wijzigingen waarbij uitwerkingen (implementaties) volgens de vorige versie van de standaard ook voldoen aan de normen/eisen van de nieuwe versie van de standaard.
- MAJOR wordt verhoogd als de nieuwe versie van de standaard zodanig wijzigt dat uitwerkingen (implementaties) volgens de vorige versie van de standaard niet meer voldoen aan de normen/eisen van de nieuwe versie van de standaard.

§ C.1.1 Patch Releases

In een patchrelease worden wijzigingen doorgevoerd die de technische specificatie niet raken. Dit kunnen tekstuele wijzigingen zijn of inhoudelijke indelingen van de documenten. De wijzigingen worden vastgelegd in release notes. Een patch releases wordt door de beheerder op eigen initiatief of op aanvragen van gebruikers doorgevoerd en gepubliceerd. Een patchrelease wordt aan het

Technisch Overleg ter kennisgeving medegedeeld. Een nieuwe patchrelease vervangt een eerdere versie in zijn geheel.

§ C.1.2 Minor releases

Een minor release geeft aan dat de nieuwe versie van de standaard zodanig is gewijzigd dat een implementatie van de voorgaande versie van de standaard voldoet aan de regels van de nieuwe versie. In een minor release kunnen wijzigingen doorgevoerd worden die de technische specificatie van een koppelvlak raken (bijvoorbeeld nieuwe functionaliteit). Voor Minor Releases wordt een uitgebreid vaststellingsprocedure gevolgd (conform het beheermodel van de standaard) en er kan in overleg met de deelnemers van het Technisch Overleg tot een migratiepad worden besloten. Dit migratiepad wordt in de release meegenomen.

§ C.1.3 Major Releases

Een major release geeft aan dat de nieuwe versie van de standaard zodanig is gewijzigd dat een implementatie van de voorgaande versie van de standaard **niet** voldoet aan de regels van de nieuwe versie. Bijvoorbeeld de overgang naar nieuwe externe (meestal internationale) standaarden binnen een bestaand profiel. Als hierbij het functionele toepassingsgebied van de standaard, waarvoor het pas-toe-of-leg-uit regime geldt, verandert, dan wordt eerst de uitgebreide vaststellingsprocedure gevolgd en vervolgens de procedure van het Forum Standaardisatie.

VOORBEELD 1: Grote wijzigingen voor Digikoppeling

Bij de overgang naar een andere externe standaard binnen een bestaand profiel kan men denken aan een overgang naar HTTP 2.0 of SOAP 1.2 binnen Digikoppeling WUS koppelvlakspecificatie. Het toevoegen van een geheel nieuw profiel kan voor Digikoppeling bestaan uit het toevoegen van een Grote Berichten Push variant of ebMS3/AS4 koppelvlakspecificaties. Deze kunnen natuurlijk bestaande koppelvlakspecificaties vervangen.

§ C.2 Toelichting en voorbeeld regels

Een versie van een standaard (versie 1.2.0) is compatible met een eerdere versie van een standaard (versie 1.1.0) als uitwerkingen/ implementaties volgens de eerdere versie 1.1.0 ook volledig voldoen aan de normen en eisen van versie 1.2.0 . Wijzigingen in de standaard kunnen impact

hebben op de technische werking van implementaties en/of op afspraken die de technische werking van implementaties niet raken bijvoorbeeld organisatorische of proces afspraken;

Voor standaarden is relevant of een realisatie volgens de oude versie van een standaard wel of niet voldoet aan de nieuwe versie van de standaard;

Voor standaarden waarbij wijzigingen op onderdelen kan verschillen tussen major en minor kan een impactmatrix opgesteld worden waarmee impact op de onderdelen gespecificeerd kan worden.

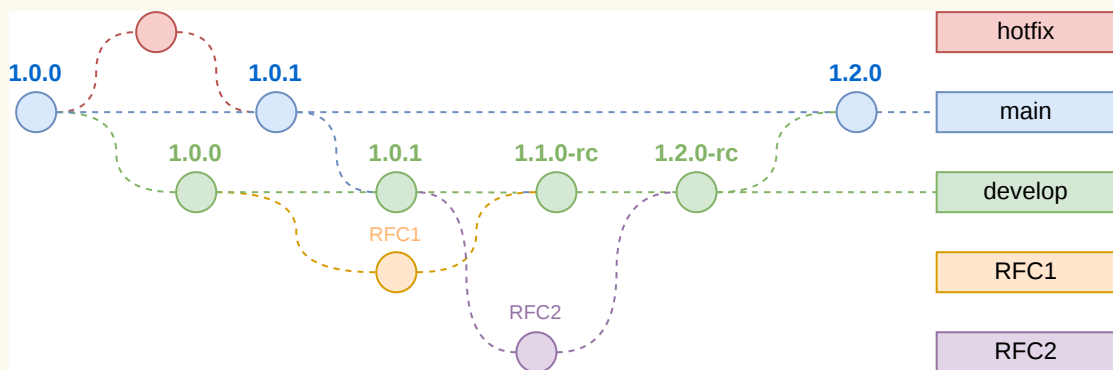
§ C.3 Versie overgangen

Wanneer een nieuwe major versie uitkomt zal de oude versie conform de afgestemde migratiepad een einddatum van geldigheid krijgen. In de overgangsperiode kunnen dus meerdere versies gepubliceerd zijn en de status geldig hebben.

Om te kunnen werken aan publicatie-, werk- en voorstelversies van documenten worden Git branches gebruikt.

VOORBEELD 2

In het onderstaande voorbeeld zien wij een standaard van 1.0.0 naar 1.1.0 ontwikkelen.



Figuur 6 Gitflow

De branch main is de huidig gepubliceerde versie en de branch develop is de werkversie. Dit wordt als documentstatus aangegeven. In de main branch staat dus een *vastgestelde versie*. Het uitwerken van een RFC gebeurt in een afsplitsing van de develop branch waarna het terug de develop branch invloeit. In het voorbeeld schema leidde RFC1 tot de eerste release candidate (rc) van versie 1.1.0 van de standaard. Wanneer de werkversie gereed en akkoord is als release stromen de wijzigingen naar de branch main. Na overgang van de develop branch naar de gepubliceerde main branch moet de status van het document worden aangepast in de main branch.

Het kan voorkomen dat gewenst wordt vlug een kleine (niet inhoudelijke) aanpassing aan de gepubliceerde versie te maken. Om bijvoorbeeld een spelfout vlug te corrigeren kan deze aanpassing op main i.p.v. develop worden uitgevoerd. In het voorbeeld leidde een hotfix tot een release van versie 1.0.1 waarna de aanpassing naar de werkversie geduwd wordt.

§ D. Lijst met figuren

[Figuur 1 BOMOS activiteitendiagram](#)

[Figuur 2 MIDO governance](#)

[Figuur 3 NL API Strategie Infographic](#)

[Figuur 4 Creative Commons Naamsvermelding 4.0 Unported licentie](#)

[Figuur 5 Behandeling van een wijzigingsvoorstel in het beheerproces](#)

[Figuur 6 Gitflow](#)

Inhoudsopgave

Status van dit document

Conformiteit

Samenvatting

1. Inleiding

1.1 Leeswijzer

1.1.1 Bijlagen

1.2 De API Standaarden

1.2.1 Nut

1.2.2 Werking

1.2.3 Status

1.3 BOMOS

2. Strategie

2.1 Visie

2.2 Governance

2.2.1 Governancestructuur

2.2.1.1 Het Kennisplatform API's (community)

2.2.1.2 Technisch Overleg

2.2.1.3 Tactisch overleg: de programmeringstafel

2.2.1.4 Het Strategisch overleg: De Programmeringsraad GDI

2.2.1.5 Overheidsbreed Beleidsoverleg Digitale Overheid (OBDO)

2.2.1.6 Architectuurraad

2.2.2 Besluitvorming

2.2.3 Deelname

2.3 Financiering

3. Tactiek

3.1 Community

3.2 Architectuur

3.2.1 Internationale, Europese en nationale standaardisatiegemeenschap

3.2.2 Samenwerking met andere beheerorganisaties

3.2.2.1 Kennisplatform API's

3.2.3 Nederlandse Overheid Referentie Architectuur (NORA)

3.3 Rechtenbeleid

3.4 Kwaliteitsbeleid en benchmarking

3.5 Adoptie en erkenning

4. Operationeel

- 4.1 Initiatie
- 4.2 Wensen en Eisen
- 4.3 Uitvoering en ontwikkeling (Wijzigingsproces)
 - 4.3.1 Wijzigingen
 - 4.3.2 Patches
 - 4.3.3 Releases
 - 4.3.4 Impact van wijzigingen en versienummering
- 4.4 Status van de standaard
- 4.5 Documentatie

5. Implementatieondersteuning

- 5.1 Opleiding en advies
- 5.2 Helpdesk
- 5.3 Validatie & Certificatie

6. Communicatie

- 6.1 Promotie
- 6.2 Publicatie
- 6.3 Klachtenafhandeling

A. Gebruik ReSpec

- A.1 Logius profiel
- A.2 Literatuurverwijzingen
- A.3 Links

B. Gebruik GitHub in het beheerproces

- B.1 Publicatie
- B.2 Wijzigingsvoorstellen
 - B.2.1 Branches
 - B.2.2 Commits op develop en main
 - B.2.3 Labels
- B.3 Automatisering en scripts

C. Versienummering

- C.1 Versioneringsmethodiek
 - C.1.1 Patch Releases
 - C.1.2 Minor releases
 - C.1.3 Major Releases
- C.2 Toelichting en voorbeeld regels
- C.3 Versie overgangen

D. Lijst met figuren

E. Index

E.1 Termen gedefinieerd door deze specificatie

E.2 Termen gedefinieerd door verwijzing

F. Contributors

G. Referenties

G.1 Normatieve referenties

§ E. Index

§ E.1 Termen gedefinieerd door deze specificatie

§ E.2 Termen gedefinieerd door verwijzing

§ F. Contributors

Logius Standaardenbeheer, Martin van der Plas en Tim van der Lippe

§ G. Referenties

§ G.1 Normatieve referenties

[RFC7763]

The text/markdown Media Type. S. Leonard. IETF. March 2016. Informational. URL:
<https://www.rfc-editor.org/rfc/rfc7763>

