Logius Standaard - Werkversie

Logboek dataverwerkingen

Logius Standaard
Werkversie 27 januari 2026

Deze versie:
https://logius-standaarden.github.io/logboek-dataverwerkingen/

Laatst gepubliceerde versie:
https://logius-standaarden.github.io/logboek-dataverwerkingen/

Laatste werkversie:
https://logius-standaarden.github.io/logboek-dataverwerkingen/

Redacteurs:
Jeroen Mulder (Ministerie van Binnenlandse Zaken en Koninkrijksrelaties)
Pieter Teekens (Ministerie van Binnenlandse Zaken en Koninkrijksrelaties)
Nil Barua (Logius)
Martin van der Plas (Logius)
Tim van der Lippe (Logius)

Auteurs:
Eelco Hotting (Ministerie van Binnenlandse Zaken en Koninkrijksrelaties)
Vedran Bilanovic (Ministerie van Binnenlandse Zaken en Koninkrijksrelaties)

Doe mee:
GitHub Logius-standaarden/logboek-dataverwerkingen
Dien een melding in
Revisiehistorie
Pull requests

Dit document is ook beschikbaar in dit niet-normatieve formaat: PDF

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.

Inhoudsopgave

§ 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.

De trefwoorden *AANBEVOLEN*, *MAG*, *MOET* en *MOETEN* in dit document moeten worden geïnterpreteerd als in BCP 14 [*RFC2119*] [*RFC8174*] als, en alleen als deze in hoofdletters zijn weergegeven, zoals hier getoond.

Context bij de standaard

De overheid wil voor burgers en bedrijven zo transparant mogelijk zijn in de omgang met hun data. Daarom is het bij de informatieverwerking in datasets belangrijk om voor elke mutatie of raadpleging vast te leggen wie deze actie wanneer uitvoert, en waarom. Voor een optimale samenwerking over organisaties en bronnen heen is voor deze logging een algemene standaard nodig.

Transparantie en verantwoording naar burgers is één van de drijfveren voor ontwikkeling van deze logging standaard maar geen onderdeel van de standaard. De standaard richt zich op vastlegging van de logging en deze eenduidige vastlegging maakt het mogelijk om, indien daar later behoefte aan is, inzage mogelijk te maken.

Verwijzingen

De Logboek Dataverwerkingen (LDV) standaard bestaat uit de volgende vier documenten:

Beschrijving van het document	Gepubliceerde versie	Werk versie	Repository
1. De LDV Normatieve Standaard	-	Logboek dataverwerkingen (werkversie)	logboek-dataverwerkingen
2. De Algemene Inleiding	-	De Algemene Inleiding (werkversie)	logboek-dataverwerkingen-inleiding
3. Het Juridische Beleidskader	-	Juridisch Beleidskader (werkversie)	logboek-dataverwerkingen-juridisch-beleidskader
4. LDV Extensie Guideline	-	Guideline voor het schrijven van een extensie voor LDV (werkversie)	logboek-extensie-template

Feedback en Issues

We moedigen gebruikers aan om meldingen of suggesties aan te maken via GitHub. Mocht dit niet mogelijk zijn, dan kunt u ook een e-mail sturen naar api@logius.nl.

1. Introductie

De standaard Logboek Dataverwerkingen beschrijft een manier om technisch interoperabele functionaliteit voor het loggen van `Dataverwerkingen` te implementeren, door voor de volgende functionaliteit de interface en het gedrag voor te schrijven:

- het vastleggen van logs van dataverwerkingen
- het aan elkaar relateren van logs van dataverwerkingen
- het aan elkaar relateren van dataverwerkingen over de grenzen van systemen

Door Dataverwerkingen te loggen volgens de standaard kunnen organisaties het datagebruik verantwoorden.

§ 1.1 Werkingsgebied van de standaard

> NOOT
>
> Deze sectie bevat de beoogde beschrijvingen voor opname op de lijst van Aanbevolen standaarden van Forum Standaardisatie

Functioneel toepassingsgebied: De standaard kan worden toegepast bij het uitvoeren van dataverwerkingen en het aan elkaar relateren van dataverwerkingen van verschillende verwerkingsverantwoordelijken.

Organisatorisch werkingsgebied: Nederlandse overheden (Rijk, provincies, gemeenten en waterschappen) en instellingen uit de (semi-) publieke sector.

§ 1.2 Doelgroep

De standaard heeft als doelgroep iedereen die zich bezighoudt met het implementeren van logging rond dataverwerkingen en beschrijft alleen wat relevant is voor de implementatie. Alle achterliggende overwegingen zijn te vinden in de Algemene inleiding en het Juridisch Beleidskader.

§ 1.3 Terminologie

De volgende lijst beschrijft terminologie in de betekenis zoals deze wordt gebruikt in dit document.

> NOOT
>
> Sommige termen zijn in bredere context al bekend, zoals Applicatie. Deze sectie geeft een vernauwende definitie van deze termen om extra eisen te stellen.

Actie

Een `Dataverwerking` bestaat uit één of meerdere kleinere discrete stappen. Een Actie is één discrete stap binnen een Dataverwerking.

Applicatie

Iedere softwaretoepassing waarmee `Dataverwerkingen` kunnen worden uitgevoerd.

Betrokkene

Als gegevens van rechtssubjecten door de overheid worden verwerkt, worden subjecten van wie de organisatie gegevens verwerkt de 'Betrokkene' genoemd. Betrokkenen in het kader van deze standaard kunnen zowel natuurlijke personen als rechtspersonen (bedrijven) zijn die met de verwerkte gegevens geïdentificeerd kunnen worden. Als identificeerbaar wordt beschouwd als een (rechts)persoon die direct of indirect kan worden geïdentificeerd, met name aan de hand van een identificerend gegeven zoals een (bedrijfs)naam, een identificatienummer, locatiedata of van een of meer elementen die kenmerkend zijn voor de fysieke, fysiologische, genetische, psychische, economische, culturele of sociale identiteit van die (rechts)persoon.

Dataverwerking

Iedere bewerking (of ieder geheel van bewerkingen) met betrekking tot gegevens, al dan niet uitgevoerd via geautomatiseerde procedures, zoals het verzamelen, vastleggen, ordenen, structureren, opslaan, bijwerken of wijzigen, opvragen, raadplegen, gebruiken, verstrekken door middel van doorzending, verspreiden of op andere wijze ter beschikking stellen, aligneren of combineren, afschermen, wissen of vernietigen van gegevens wordt opgevat als een Dataverwerking. Iedere Dataverwerking bestaat uit één of meerdere `Acties`.

Inzage

De `Betrokkene` heeft het recht om van de `Verantwoordelijke` uitsluitsel te verkrijgen over het al dan niet verwerken van hem betreffende gegevens en, wanneer dat het geval is, om inzage te verkrijgen van die gegevens. Voor natuurlijke personen sluit dit aan bij hun recht op inzage zoals bedoeld in de AVG, voor rechtspersonen([*AVG*], art. 15, lid 1). Voor rechtspersonen sluit dit aan bij het recht op transparantie over besluitvorming waar zij bij betrokken zijn. Met *Inzage* doelen we op de handeling waarmee uitvoering wordt gegeven aan dat recht.

Logboek

Softwaretoepassing waarmee het log van `Dataverwerkingen` kunnen worden bijgehouden.

Logregel

Resultaat van een enkele gebeurtenis in de logging.

Register

Register waarin statische data over `Verwerkingsactiviteiten` kunnen worden geregistreerd en ter beschikking gesteld. Daarin wordt onder meer het doel van de verwerkingen opgenomen, zoals dit bijvoorbeeld op grond van de AVG voor verwerkingen van persoonsdata al verplicht is. Registers in het kader van de standaard beogen een bredere reikwijdte dan persoonsdata.

> VOORBEELD 1
>
> Het Register van Verwerkingsactiviteiten (RvVA in het kader van de AVG) en het Algoritmeregister zijn voorbeelden van `Registers`.

Trace

Concept waarmee bij elkaar behorende `Dataverwerkingen` binnen de grenzen van een systeem worden gegroepeerd.

Verwerkingsverantwoordelijke

Een natuurlijke persoon of rechtspersoon, een overheidsinstantie, een dienst of een ander orgaan die/dat, alleen of samen met anderen, het doel van en de middelen voor de verwerking van data vaststelt. Deze definitie is gebaseerd op ([*AVG*] art. 4, lid 7.), maar laat de verantwoordelijkheid betrekking hebben op de verwerking van álle data, niet alleen persoonsdata.

> NOOT
>
> In de standaard wordt de Verwerkingsverantwoordelijke aangeduid als de Verantwoordelijke voor de leesbaarheid.

Verwerker

Een natuurlijke persoon of rechtspersoon, een overheidsinstantie, een dienst of een ander orgaan die/dat ten behoeve van de `Verwerkingsverantwoordelijke` persoonsdata verwerkt. Deze definitie is gebaseerd op ([*AVG*] art. 4, lid 8.), maar laat de verantwoordelijkheid betrekking hebben op de verwerking van álle data, niet alleen persoonsdata.

Verwerkingsactiviteit

Activiteiten die een organisatie onderkent heeft als activiteiten waarbinnen `Dataverwerkingen` plaatsvinden.

§ 1.4 Algemene werking van de standaard

`Applicaties` loggen metadata over `Dataverwerkingen` in een daarvoor ingerichte softwaretoepassing, het `Logboek`. Elke Dataverwerking wordt apart gelogd. Dataverwerkingen binnen dezelfde context (bijvoorbeeld een organisatie of een verantwoordelijkheid binnen een organisatie) worden gegroepeerd met behulp van een Trace. Wanneer een Dataverwerking een andere Dataverwerking tot gevolg heeft worden de logregels van beide Dataverwerkingen aan elkaar gelinkt. Statische informatie over Dataverwerkingen kan worden opgezocht in Registers op basis van een verwijzing die in elke logregel wordt opgenomen.

§ 1.4.1 Extensies

De standaard Logboek Dataverwerkingen specificeert de basis voor het loggen en aan elkaar relateren van Dataverwerkingen. Aanvullende functionaliteit wordt gestandaardiseerd in `extensies`, conform de *LDV Extensie Guideline* richtlijnen. Enkele voorbeelden van extensies (deze zijn nog niet per definitie vastgesteld):

- Extensie (geo)objecten
  Deze extensie specificeert hoe dataverwerkingen voor (geo)objecten kunnen worden vastgelegd en beheerd in een logboek.

- Concept-extensie zorg
  Deze extensie was een proof-of-concept hoe dataverwerkingen die aan de [*NEN7513*] norm kunnen worden vastgelegd en beheerd in een logboek.

- Extensie lezen
  Deze extensie specificeert hoe vanuit een logboek de data gelezen kan worden.

§ 1.4.2 Profielen

In een profiel worden aanvullende beperkingen en verplichtingen vastgelegd over het gebruik van de standaard. Op deze manier kan een groep organisaties interoperabiliteit organiseren. Voorbeelden van aanvullende afspraken in een profiel zijn:

- De combinatie van extensies die gebruikt wordt
- Afspraken over specifieke aanvullende eisen (bijvoorbeeld over TLS configuratie)

- Afspraken over data-retentie
- De wijze waarop pseudonimisering van persoonsdata plaatsvindt

1.4.3 Use case

Een typische use case voor het gebruik van de standaard is een samenwerking tussen meerdere organisaties die interoperabiliteit willen bereiken bij het loggen van `Dataverwerkingen`, om zo op eenduidige manier te kunnen verantwoorden over de dataverwerking.

2. Architectuur

Deze sectie beschrijft de algemene architectuur voor het loggen van dataverwerkingen bij toepassing van deze standaard.

2.1 Context

Op hoog abstractieniveau zijn voor het begrijpen van deze standaard de volgende componenten te onderscheiden:

- `Applicatie`
- `Logboek`
- `Register`

Applicaties schrijven logs over Dataverwerkingen weg in een Logboek. Logregels in het Logboek verwijzen naar nadere informatie in een Register.

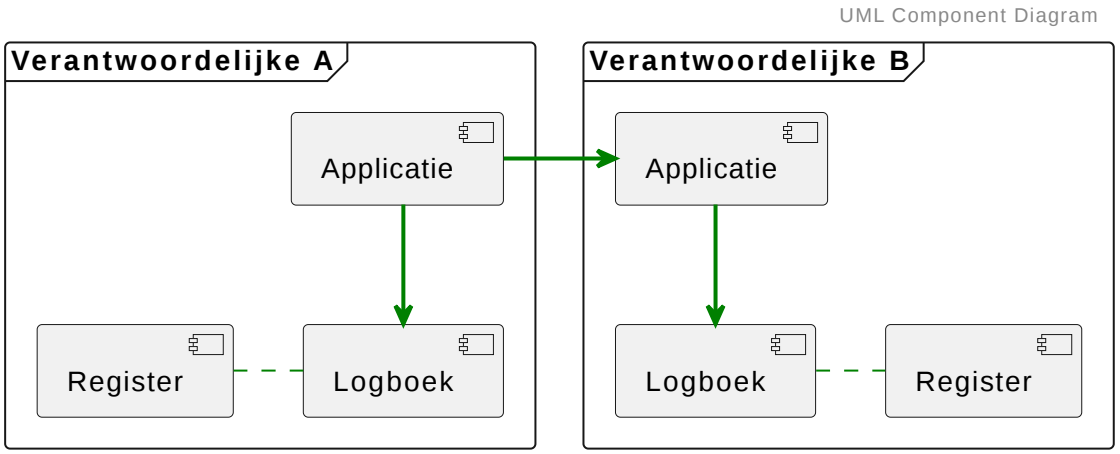
Een Dataverwerking kan plaatsvinden over de grenzen van een verantwoordelijkheid. In dat geval roept een Applicatie van Verantwoordelijke A de Applicatie van Verantwoordelijke B aan. Denk bijvoorbeeld aan het bevragen of muteren van data via een Application Programming Interface (API).

Een Verantwoordelijke is bijvoorbeeld een organisatie, maar kan ook bestaan uit meerdere organisaties die allemaal onder dezelfde Verantwoordelijke werk uitvoeren. Denk daarbij aan Verwerkers in het kader van de AVG.

Iedere Verantwoordelijke kan een veelheid aan Applicaties, Logboeken en Registers gebruiken. Iedere Verantwoordelijke houdt alleen Logregels bij over eigen Dataverwerkingen. Op basis van

metadata die tussen Applicaties wordt uitgewisseld is het mogelijk om bij elkaar behorende Logregels in meerdere Logboeken aan elkaar te relateren.

Registers bevatten statische informatie waar vanuit Logregels naar verwezen kan worden voor extra informatie over een Dataverwerking.



Figuur 1 Componenten in context

De standaard beschrijft de interfaces (in het diagram aangeduid met groene lijnen), en het gedrag van de componenten voor zover relevant om technisch interoperabel te worden.

De relatie tussen Logboek en Registers is los. Een Register hoeft niet digitaal te bestaan, wel moet een relatie gelegd kunnen worden vanuit de logregels in het Logboek naar aanvullende data in Registers die de Logregels van nedere context voorzien.

§ 2.2 Componenten

Deze sectie beschrijft de verschillende componenten van de standaard. Tevens is er een canoniek gegevensmodel (SVG, XLSX) voor een uniforme structuur en terminologie voor alle relevante data die vastgelegd wordt in de verschillende componenten.

§ 2.2.1 Applicatie

Een `Applicatie` is een softwarecomponent of groep van softwarecomponenten waarmee een Dataverwerking wordt uitgevoerd. Een Applicatie kan in allerlei vormen voorkomen. Voor de architectuur is niet relevant welke vorm de Applicatie heeft, het is slechts relevant dat dit de component is waar een Dataverwerking wordt uitgevoerd.

In een Applicatie is de context van de Dataverwerking bekend, zoals welke Verwerkingsactiviteit wordt uitgevoerd met de Dataverwerking. Het is dan ook de Applicatie die het loggen van de Dataverwerking initiëert.

§ 2.2.2 Logboek

Een `Logboek` is een Applicatie met een specifieke rol in de context van deze standaard. In het Logboek worden Dataverwerkingen gelogd.

Dataverwerkingen in het Logboek zelf worden *niet* gelogd in een Logboek Dataverwerkingen, dit zou een oneindige recursiviteit veroorzaken.

§ 2.2.3 Register

Een `Register` bevat statische informatie over Dataverwerkingen. Elk record in een Register heeft een unieke identificatiecode waarmee de Verwerkingsactiviteit kan worden aangeduid. Deze identificatiecode wordt gebruikt om vanuit een Logregel te linken naar aanvullende informatie in een Register.

Het Register kan een Applicatie zijn, in dat geval is het een Applicatie met een specifieke rol in de context van deze standaard. Eventueel kan het ook een Register in de vorm van een document zijn.

Dataverwerkingen in het Register worden gelogd in een Logboek Dataverwerkingen.

Voor alle Dataverwerkingen waarbij persoonsdata worden verwerkt is wettelijk geregeld dat de Verwerkingsactiviteiten moeten worden beschreven in het zogenaamde Register van Verwerkingsactiviteiten (AVG art. 30). Dit Register wordt verondersteld aanwezig te zijn in iedere organisatie die de standaard Logboek Dataverwerkingen toepast.

Verwerkingsactiviteiten waarin geen persoonsdata worden verwerkt staan niet verplicht in het Register van Verwerkingsactiviteiten. De standaard laat ruimte om dit op te lossen naar eigen voorkeur:

- In het bestaande Register ook Verwerkingsactiviteiten opnemen zonder persoonsdata, al is dit niet wettelijk verplicht
- Zelf een ander Register opzetten met gelijke interface maar specifiek voor Verwerkingsactiviteiten zonder persoonsdata

Het is daarnaast ook mogelijk om Registers te gebruiken met heel andere statische informatie die meer context geeft over een Logregel, bijv. informatie over de gebruikte beslisregels of van

toepassing zijnde normen. Dit wordt niet verder uitgewerkt.

> NOOT
>
> Op dit moment is er geen specificatie voor het ontsluiten van een `Register` met een API. Hier zal een toekomstige versie van de standaard wel in voorzien.

§ 2.3 Scope

In deze sectie wordt de scope van de standaard afgebakend.

§ 2.3.1 Vastlegging door Verantwoordelijke

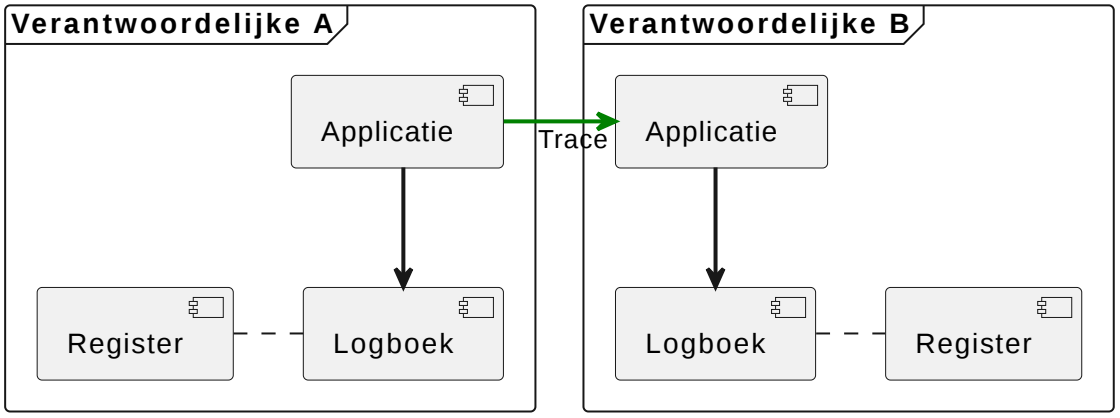
Voor een juiste toepassing van de standaard is het nodig om strict de grenzen aan te houden die passen bij de Verantwoording die een `Verantwoordelijke` af moet kunnen leggen. Het wordt *AANBEVOLEN* om alle Dataverwerkingen te loggen alsof zij persoonsdata bevatten, ook wanneer de Dataverwerking geen persoonsdata betreft. Dit omdat het wettelijk kader dat leidt tot verantwoordingsplicht breder is dan alleen de AVG. Logregels kunnen ook worden gebruikt voor bijvoorbeeld het verantwoorden welke data gebruikt zijn bij het nemen van een besluit.

Belangrijk uitgangspunt is dat een Verantwoordelijke alleen Logregels bijhoudt voor Dataverwerkingen die onder eigen verantwoordelijkheid plaatsvinden.

Een zogenaamde `Verwerker` die Dataverwerkingen uitvoert in opdracht van een Verantwoordelijke wordt in deze standaard beschouwd als deel van de Verantwoordelijke. Van welke Logboeken en Registers een Verwerker gebruik maakt is een implementatiekeuze.

§ 2.3.2 Geen inhoudelijke uitwisseling tussen Verantwoordelijken

Er wordt met de standaard geen inhoudelijke informatie over Dataverwerkingen uitgewisseld tussen Verantwoordelijken. Dit is niet nodig, aangezien iedere Verantwoordelijke alleen Logregels over eigen Dataverwerkingen vastlegt. De informatie die wordt uitgewisseld is beperkt tot zogenaamde `Trace`-informatie waarmee Logregels van de ene Verantwoordelijke gerelateerd kunnen worden aan Logregels bij de andere Verantwoordelijke.



Figuur 2 Context Dataverwerking meegeven over Grenzen

§ 2.3.3 Geen specificatie voor het beheren van Logboeken

De standaard specificeert een interface voor het wegschrijven van Logregels. Dit is het deel dat in alle organisaties hetzelfde moet zijn om interoperabel te zijn. Het beheren van een Logboek is vrij in te vullen per implementatie.

Dit betekent o.a. dat de standaard *geen* gedrag of interfaces specificeert voor:

- het verwijderen of muteren van Logregels
- het regelen van toegang tot het Logboek
- het regelen van duurzame toegankelijkheid
- het regelen van archivering en verwijdering van Logregels

§ 2.3.4 Geen data over gebruikers in Logregels

In Logregels ligt geen informatie vast over welke specifieke medewerker van de Verantwoordelijke ofwel de gebruiker de Dataverwerking heeft uitgevoerd. Deze informatie hoort niet in het Logboek maar in een auditlog, en is daarmee buiten scope van de standaard. Wel is het mogelijk om vanuit auditlogs de relatie te leggen naar specifieke Logregels in het Logboek. Voor meer informatie over deze kwestie, zie het besluit over gebruikers..

§ 3. Specificaties

Deze sectie geeft de specificatie voor de te gebruiken protocollen en interfaces en het verwachte gedrag van de componenten.

§ 3.1 Protocollen

De protocollen die worden gebruikt tussen applicatie en logboek en voor het uitvoeren van transacties tussen applicaties worden niet voorgeschreven in de standaard.

> NOOT
>
> Let wel, met "de protocollen" bedoelen we de manier van afleveren van berichten tussen de componenten. Deze standaard beschrijft wel degelijk interfaces van de berichten zelf waar de componenten aan *MOETEN* voldoen, met als doel interopabiliteit tussen functionaliteit van componenten. De standaard laat vrij hoe die informatie tussen componenten wordt doorgegeven, omdat dat afhangt van de technische/architecturele keuzes die software ontwikkelaars maken. Dit biedt de vrijheid om de standaard toe te voegen aan vrijwel iedere softwareoplossing.

Het is *AANBEVOLEN* om het OpenTelemetry Protocol (OTLP) te gebruiken in de interactie tussen Applicatie en Logboek.

> NOOT
>
> OpenTelemetry is een standaard en open source framework voor het beheren, genereren, verzamelen en exporteren van telemetriedata. Door het gebruik van deze open standaard kunnen leverancierspecifieke integraties voorkomen worden. OpenTelemetry is een CNCF incubating project.

Als gebruik wordt gemaakt van HTTP/1.1 [*RFC9112*] of HTTP/2 [*RFC9113*] voor het uitvoeren van dataverwerkingen in meerdere applicaties *MOET* gebruik worden gemaakt van de *Trace Context* specificatie voor het uitwisselen van metadata over `Traces`.

§ 3.2 Component: Logboek

Voor ieder Logboek waarin Dataverwerkingen worden gelogd gelden de volgende specificaties voor gedrag en interface.

§ 3.2.1 Gedrag

Het Logboek *MOET* TLS afdwingen op connecties volgens de binnen de organisatie gangbare standaard.

Het Logboek *MOET* het wegschrijven van elke logregel bevestigen.

§ 3.2.2 Interface

De interface *MOET* de volgende velden implementeren:

> NOOT
>
> In SDK's van OpenTelemetry worden soms andere structuren of capitalization gebruikt dan hoe die in deze tabel voorkomen. Dit komt doordat programmeertalen verschillende naming conventions gebruiken en de SDK's deze conventions volgen. OpenTelemetry enforceert zelf ook geen eenduidige naamgeving.
>
> Voor het wegschrijven van logregels is dit geen probleem, zolang elk veld duidelijk te herleiden is bij het gebruik van de SDK. Voor het lezen van logregels (wat in een aparte toekomstige extensie wordt gestandaardiseerd) is dit wel van belang. In die extensie zal er wel een specifieke structuur beschreven en hoe daar aan kan worden voldaan.

Veld	Type	Verplicht?
`trace_id`	16 byte	verplicht
`span_id`	8 byte	verplicht
`status`	enum	verplicht
`name`	string	verplicht
`start_time`	uint64	verplicht
`end_time`	uint64	verplicht
`parent_span_id`	8 byte	optioneel
`resource`	object	optioneel
`attributes`	object	verplicht

§ *3.2.2.1 `trace_id`*

Unieke identificerende code van `Trace` die `Dataverwerking` volgt. Als er meerdere applicaties dataverwerkingen uitvoeren ten behoeve van 1 originele dataverwerking, dan is de trace code identiek voor al deze dataverwerkingen, zie gedrag van applicaties.

§ *3.2.2.2 `span_id`*

Unieke identificerende code van `Actie` binnen de `Dataverwerking`. Een applicatie kan meerdere `span_id` voor dezelfde `trace_id` hebben.

§ *3.2.2.3 `status`*

Het veld `status` is een enumeratie die de volgende waarden kan bevatten:

- `Unset`: De standaardwaarde voor elke `status` is `Unset`. Dit betekent dat de dataverwerking is uitgevoerd zonder interne fout. Deze waarde wordt toegepast wanneer de dataverwerking technisch correct is afgerond, ook als er geen resultaat beschikbaar is of wanneer de invoer onvolledig was.
- `Ok`: De waarde `Ok` kan optioneel gebruikt worden wanneer de ontwikkelaar expliciet wil markeren dat de dataverwerking succesvol is afgerond. Dit is afhankelijk van hoe de organisatie die de standaard implementeert een dataverwerking als succesvol definieert en of zij dit onderscheid expliciet willen loggen als andere waarde dan `Unset`.
- `Error`: De waarde `Error` wordt toegekend bij fouten die zijn ontstaan binnen het systeem dat de dataverwerking uitvoert, zoals interne fouten of mislukte uitvoeringen door technische oorzaken.

De waarden `Unset` en `Ok` worden altijd bepaald op basis van het resultaat van de verwerking. De waarde `Ok` is optioneel en kan gebruikt worden als de organisatie ervoor kiest dataverwerkingen expliciet als succesvol te markeren. `Error` is alleen nodig als er een fout is opgetreden bij het interne proces. Een dataverwerking die niet klopt op basis van de gegeven gebruikersinput, maar die zonder fouten is afgehandeld, hoort dus status `Unset` te krijgen.

§ *3.2.2.4 `name`*

Naam van de specifieke `Actie` binnen de `Dataverwerking`. Dit is een tekstuele beschrijving bestemd voor mensen, niet voor machines.

§ *3.2.2.5 `start_time`*

Tijdstip waarop de `Actie` gestart is in milliseconden sinds Epoch.

§ *3.2.2.6 `end_time`*

Tijdstip waarop de `Actie` beëindigd is in milliseconden sinds Epoch.

§ *3.2.2.7 `parent_span_id`*

Unieke identificerende code aanroepende `Actie`. Dit geldt voor zowel binnen de huidige applicatie als bij een aanroep van een andere applicatie. Als `dpl.core.foreign_operation.processor` aanwezig is (zie attributes), dan is het een aanroep van een andere applicatie.

§ *3.2.2.8 `resource`*

Het veld `resource` is een object, opgebouwd uit de volgende velden:

Veldnaam	Type	Omschrijving
attributes	Any	Een object met velden dat gebruikt wordt om een systeem, applicatie of component aan te duiden op een manier die binnen de organisatie gebruikelijk is. Denk hierbij aan velden als naam en versienummer van een applicatie, of een verwijzing naar een record in een CMDB.

VOORBEELD 2

```
{
  "attributes": {
    "dpl.core.processing_activity_id": 14
  },
  "resource": {
    "attributes": {
      "process.pid": 12345,
      "process.executable.name": "node",
      "process.command": "/app.js",
      "process.command_line": "/bin/node /app.js",
      "process.runtime.version": "16.17.0",
      "process.runtime.name": "nodejs",
      "process.runtime.description": "Node.js"
    }
  }
}
```

NOOT

Dit is een veld wat een object is, met daarin een veld "attributes". Deze structuur komt voor uit OpenTelemetry (Resource definitie). Ook al is OpenTelemetry niet verplicht te gebruiken, de structuur van "resource" is er wel op gebaseerd. Een logregel bevat de hoofdvelden "resource" en "attributes" (volgende sectie) en die dienen verschillende doelen.

§ *3.2.2.9 `attributes`*

Het veld `attributes` is een object, opgebouwd uit velden in een namespace met prefix `dpl` (data processing log). De volgende velden zijn vereist in de namespace `core`:

Veldnaam	Type	Omschrijving
dpl.core.processing_activity_id	URI	Verwijzing naar een Register met meer informatie over de Verwerkingsactiviteit.
dpl.core.data_subject_id	String	Unieke, versleutelde identificerende code van de Betrokkene.
dpl.core.data_subject_id_type	String	Type van de identificerende code, zoals BSN, personeelsnummer, of een URI naar een Register dat

Veldnaam	Type	Omschrijving
		het type specificeert.

De volgende velden in de namespace `core` zijn enkel vereist als er een aanroepende Applicatie is, zie de specificatie van het gedrag van Applicaties.

Veldnaam	Type	Omschrijving
dpl.core.foreign_operation.processor	URL	Link naar externe applicatie

> NOOT
>
> Extensies mogen attributen in andere namespaces definieren. Hiervoor gelden de *LDV Extensie Guideline* richtlijnen. Extensies moeten vastgesteld zijn, alvorens een attribuut mag worden gebruikt. Dit om te voorkomen dat niet-gestandaardiseerde namespaces worden gebruikt en er geen eenduidig gebruik van attributen ontstaat.

§ 3.3 Component: Applicatie

Voor iedere `Applicatie` waarin Dataverwerkingen plaatsvinden gelden de volgende specificaties voor gedrag.

§ 3.3.1 Gedrag van Applicatie

Het gespecificeerde gedrag van Applicaties is erop gericht om de interface van het Logboek te gebruiken. Voor alle metadata geldt dat de specificatie te vinden is in de interface van het Logboek.

De Applicatie *MOET* een nieuwe Trace met een uniek `trace_id` bijhouden voor iedere nieuwe Dataverwerking. In een Trace wordt de metadata bijgehouden die nodig is om de interface van een Logboek te gebruiken.

Een Dataverwerking kan uit meerdere acties bestaan. De applicatie *MOET* een voor iedere nieuwe actie een unieke `span_id` bijhouden. Iedere Trace heeft tenminste één `span_id`.

Wanneer een actie binnen een Applicatie is gestart door een andere actie, dan *MOET* de Applicatie de `trace_id` ongewijzigd overnemen en de `span_id` opnemen in een veld genaamd `parent_span_id` voor deze nieuwe actie.

Als een Dataverwerking meerdere Betrokkenen heeft dan *MOET* de applicatie voor iedere Betrokkene een aparte logregel wegschrijven. Een logregel kan naar 0 of 1 Betrokkenen verwijzen.

De Applicatie *MOET* voor iedere actie (`span_id`) een logregel wegschrijven via de interface van het Logboek.

De Applicatie *MOET* bijhouden of een actie geslaagd of mislukt is en dit per Dataverwerking als status (`status`) meegeven in de Logregel.

Als de Applicatie een verzoek van een andere Applicatie kan ontvangen, *MOET* de Applicatie metadata volgens de W3C Trace Context standaard kunnen verwerken en gebruiken in de eigen Trace(s). Metadata verkregen via W3C Trace Context *MOET* in `attributes` meegenomen worden als velden die beginnen met `dpl.core.foreign_operation`. Zie de specificatie van attributes in het logboek voor de lijst van velden.

Als de Applicatie een verzoek aan een andere Applicatie kan versturen, *MOET* de Applicatie metadata volgens de W3C Trace Context standaard meegeven aan dit verzoek.

De Applicatie *MAG* NIET gebruik maken van *Log Sampling*.

§ *3.3.1.1 Loggen van Dataverwerkingen met persoonsdata*

Voor iedere Betrokkene moet iedere Dataverwerking apart gelogd worden. De Applicatie *MOET* in elke Logregel een identificerende code van de Betrokkene opnemen in `dpl.core.data_subject_id` en aan te duiden welk soort identificerende code wordt gebruikt in `dpl.core.data_subject_id_type`. Het wordt *AANBEVOLEN* om de identificerende code te pseudonimiseren.

Wanneer een enkele Dataverwerking meerdere Betrokkenen heeft, *MOET* de Applicatie voor elke Betrokkene een nieuwe actie met unieke `span_id` starten en deze onder de reeds bekende actie voegen door het `span_id` daarvan op te nemen als `parent_span_id` in de nieuwe actie. Voor iedere betrokkene wordt een *child operation* bijgehouden.

Let op: het kan zijn dat pas na een antwoord van een externe Applicatie bekend is dat er meerdere Betrokkenen zijn bij een Dataverwerking, in dat geval moeten na ontvangst van het antwoord de nieuwe acties ten behoeve van correcte logging gestart worden.

Iedere Dataverwerking van persoonsdata betreft een Verwerkingsactiviteit die in het Register van Verwerkingsactiviteiten moet zijn opgenomen. De Applicatie *MOET* in de Logregel een verwijzing naar de juiste Verwerkingsactiviteit in het Register van Verwerkingsactiviteiten opnemen in het veld `dpl.core.processing_activity_id`.

§ *3.3.1.2 Loggen van Dataverwerkingen zonder data*

Dataverwerkingen zonder persoonsdata zijn over het algemeen niet als Verwerkingsactiviteit opgenomen in het Register van Verwerkingsactiviteiten. Het wordt aanbevolen om wel een soortgelijk register bij te houden voor alle Dataverwerkingen zonder persoonsdata.

Het wordt *AANBEVOLEN* dat de Applicatie in de Logregel een verwijzing naar de juiste Verwerkingsactiviteit in een daarvoor aan te wijzen Register opneemt in het veld `dpl.core.processing_activity_id`.

§ **3.3.2 Foutafhandeling**

Fouten kunnen in iedere applicatie optreden. Fouten kunnen ontstaan door bijvoorbeeld verkeerde invoer door de gebruiker, een fout in de software van de applicatie of een connectie met een andere applicatie die niet werkt. Deze sectie geeft een handreiking ten aanzien van de afhandeling van foutsituaties met betrekking tot het gebruik van het Logboek Dataverwerkingen.

> NOOT
>
> Let op: wanneer een gebruiker een verwerking bewust afbreekt, wordt dit niet als een fout beschouwd. Het is aan te raden om dit als een expliciete stap in het proces op te nemen, zodat ook deze handeling kan worden gelogd. In zulke gevallen is de `status` van de verwerking `Ok`, omdat er sprake is van een verwachte en correcte actie van de gebruiker.

§ *3.3.2.1 Uitgangspunten registratie foutsituaties*

De volgende punten zijn belangrijk in het ontwerpen en implementeren van de registratie van foutsituaties in relatie tot het Logboek Dataverwerkingen:

- Gebruik zoveel mogelijk de standaardfoutmethodes van de gebruikte ontwikkeltaal en/of SDKs.
- Foutdata moeten worden gerelateerd aan een `trace_id` en `span_id`.
- De software van de applicatie die de registratie van de logdata registreert, moet er voor zorgen dat er geen fout optreedt in 'run-time'. Bijvoorbeeld als `name` leeg is, moet deze automatisch worden gevuld met een waarde zodat er in ieder geval op dit punt geen fout kan optreden.

§ *3.3.2.2 Locatie van opslag*

De foutsituatie kan zowel in het Logboek als in een extern component worden registreerd. Beiden hebben voor- en nadelen:

Locatie	Voordelen	Nadelen
In het logboek	• Fouten zijn te herkennen door `status` (=error). • Fouten worden apart geregistreerd als transactie, waardoor succesvolle en gefaalde transacties aan elkaar gerelateerd kunnen worden.	• Er moet een trigger zijn, zodat de beheerder ingelicht wordt dat er een foutsituatie is ontstaan. • Als er een grote hoeveelheid logregels zijn, kost het zoeken meer computatiewerk.
In een extern component	• Alle foutsituaties staan gecentraliseerd opgeslagen waardoor monitoring op fouten eenvoudiger is.	• Foutsituaties moeten worden geregistreerd inclusief `trace_id` en `span_id`. • Extra inspanning om de foutsituatie later te relateren.

§ *3.3.2.3 Attributes*

Specifieke foutdata worden opgeslagen als velden in `attributes`:

Veldnaam	Type	Omschrijving
exception.message	String	Tekstuele beschrijving van de fout
exception.type	String	Type foutmelding (idealiter een dynamische foutmelding)
exception.stacktrace	String	Volledige stacktrace (als dat mogelijk is, afhankelijk van programmeertaal)

VOORBEELD 3

Een foutregistratie kan er als volgt uitzien:

```
{
  "trace_id": "7bba9f33312b3dabc8f8e90c7c61f194",
  "span_id": "2a3f5c8d1e6b4a09",
  "status": "error",
  "name": "Database connection failure",
  "start_time": "2025-03-09T20:21:00Z",
  "end_time": "2025-03-09T20:23:00Z",
  "parent_span_id": "",
  "attributes": {
    "exception.message": "HTTP 500 error processing /api/v1/orders",
    "exception.type": "TimeoutException",
    "exception.stacktrace": "TimeoutException: Database connection fai
  }
}
```

3.4 Component: Register

Voor ieder `Register` met statische data over Dataverwerkingen gelden de volgende specificaties voor het gedrag en de interface.

3.4.1 Gedrag van Register

Het Register *MOET* iedere relevante wijziging van een Verwerkingsactiviteit opslaan als een nieuwe versie met tijdstip, zodat de `dpl.core.processing_activity_id` naar een eenduidige versie van de verwerkingsactiviteit verwijst in combinatie met het tijdstip.

4. Detailniveaus

Logging kan op verschillende ***Detailniveaus***: hoe hoger het detailniveau, hoe gedetailleerder er wordt gelogd.

§ 4.1 Definities van niveaus

We maken drie verschillende niveaus op. Ter verduidelijking van elk niveau gebruiken we een niet-normatief voorbeeld van het opvragen van informatie over een auto.

§ 4.1.1 Niveau 1: registerverwijzing

Op het laagste niveau wordt er enkel naar het `Register` verwezen. Dit betekent dat de `Logregel` enkel de referentie naar het register bevat. Hieruit kan worden opgemaakt welke potentiële data is verwerkt, maar hiermee kan niet worden herleid welke data is verwerkt op basis van enkel de Logregel. Dit kan wel op basis van de verwerkingsactiviteit ID.

Voorbeeld referentie: er is informatie over het bezit van een auto opgevraagd. Dit register bevat informatie zoals BSN, Adres, Kenteken, Lease-contract, APK gekeurd. Er wordt hier niet verder gespecificeerd welke categorieën van informatie zijn opgevraagd. De `Logregel` bevat dus "Bezit van auto opgevraagd uit RDW-register: `dpl.core.processing_activity_id` <<URI>>"

§ 4.1.2 Niveau 2: kolomverwijzing

Op dit niveau wordt er zowel naar het betreffende `Register` verwezen alsmede de specifieke categorieën van data die zijn verwerkt. Hieruit kan dus worden opgemaakt welke specifieke categorieën van data zijn verwerkt, maar kan er niet worden herleid wat de waarde van de data is tijdens het loggen.

Voorbeeld referentie: er is informatie over het bezit van een auto opgevraagd. Dit register bevat informatie zoals BSN, Adres, Kenteken, Lease-contract, APK gekeurd. De `Dataverwerking` betrof het BSN, Kenteken en APK gekeurd. Omdat de overige categorien niet werden gebruikt, zijn die ook niet gelogd. De `Logregel` bevat dus "Bezit van auto opgevraagd uit RDW-register: BSN, Kenteken, APK gekeurd, `dpl.core.processing_activity_id` <<URI>>"

§ 4.1.3 Niveau 3: concrete data

Op het hoogste niveau wordt alle data in de `Logregel` gezet. Dit betekent dat alle relevante categorieën uit het `Register` met bijbehorende concrete data worden gelogd. Op dit niveau kan de

gehele `Dataverwerking` worden gereconstrueerd.

Voorbeeld referentie: er is informatie over het bezit van een auto opgevraagd. Dit register bevat informatie zoals BSN, Adres, Kenteken, Lease-contract, APK gekeurd. De `Dataverwerking` bevat de BSN, Kenteken en APK gekeurd met specifieke data. De `Logregel` bevat dus "Bezit van auto opgevraagd uit RDW-register: BSN 1234, Kenteken 1-ABC-23, APK gekeurd: ja, `dpl.core.processing_activity_id` <<URI>>"

§ 4.2 Implicaties van niveaus

Hoe hoger het niveau, hoe bruikbaarder de data is die uit het `Logboek` kan worden opgemaakt. Echter, de consequenties van een hoger niveau brengen ook lastigere vraagstukken met zich mee omtrent schaalbaarheid en technische haalbaarheid. Als voor elke `Dataverwerking` alle informatie wordt gelogd, dan kan dat problemen opleveren voor de hoeveelheid data en ook of het doenlijk is om de data te verwijderen als de `Betrokkene` daar om vraagt.

Tegelijkertijd is het laagste niveau van logging niet per definitie voldoende om de vereiste verantwoordelijkheid af te kunnen leggen. Dit hangt af van de situatie en de wettelijke bepalingen die verbonden zijn aan een `Dataverwerking`. Daarom is het van belang dat bij elke dataverwerking er wordt gekeken welk niveau van toepassing is, in plaats van een generiek niveau voor alle dataverwerkingen te bepalen. Hierbij is het niet noodzakelijk dat altijd het meest gedetailleerde niveau wordt gekozen.

§ 5. Lijst met figuren

§ A. Index

§ A.1 Begrippen gedefinieerd door deze specificatie

Inzage §1.3

Logboek §1.3

Logregel §1.3

Register §1.3

Trace §1.3

Verwerker §1.3

Verwerkingsactiviteit §1.3

Verwerkingsverantwoordelijke §1.3

§ A.2 Begrippen gedefinieerd door verwijzing

§ B. Referenties

§ B.1 Normatieve referenties

[AVG]

Algemene Verordening Gegevensbescherming. Verordening (EU) 2016/679 van het Europees Parlement. 27 april 2016. URL: https://eur-lex.europa.eu/legal-content/NL/TXT/?uri=CELEX%3A32016R0679

[logboek-extensie-guideline]

LDV Extensie Guideline. Logius. URL: https://logius-standaarden.github.io/logboek-extensie-template/

[RFC2119]

Key words for use in RFCs to Indicate Requirement Levels. S. Bradner. IETF. March 1997. Best Current Practice. URL: https://www.rfc-editor.org/rfc/rfc2119

[RFC8174]

Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words. B. Leiba. IETF. May 2017. Best Current Practice. URL: https://www.rfc-editor.org/rfc/rfc8174

[RFC9112]

HTTP/1.1. R. Fielding, Ed.; M. Nottingham, Ed.; J. Reschke, Ed. IETF. June 2022. Internet Standard. URL: https://httpwg.org/specs/rfc9112.html

[RFC9113]

HTTP/2. M. Thomson, Ed.; C. Benfield, Ed. IETF. June 2022. Proposed Standard. URL: https://httpwg.org/specs/rfc9113.html

[trace-context-1]

Trace Context. Sergey Kanzhelev; Morgan McLean; Alois Reitbauer; Bogdan Drutu; Nik Molnar; Yuri Shkuro. W3C. 23 November 2021. W3C Recommendation. URL: https://www.w3.org/TR/trace-context-1/

§ B.2 Informatieve referenties

[NEN7513]

Medische informatica - Logging - Vastleggen van acties op persoonlijke gezondheidsinformatie. Normcommissie Informatievoorziening in de zorg. NEN. December 2024. URL: https://www.nen.nl/nen-7513-2024-nl-329182

↥