Ministerie van Volksgezondheid,
Welzijn en Sport

# Open source ambitieladder in maatwerk aanbesteding of opdracht

Maurice Hendriks
In samenwerking met alle bijdragers

Gepubliceerd op 14 september 2025

# Inhoud

## Introductie

Dit document biedt met de 'open source ambitieladder' een praktisch hulpmiddel voor aanbestedende diensten die een opdracht voor maatwerksoftware of dienstverlening in de markt zetten. De ladder helpt je om de gewenste mate van 'open source' te bepalen en te specificeren, afhankelijk van de strategische doelen. Denk hierbij aan doelstellingen zoals het vergroten van de transparantie, het bevorderen van hergebruik of het waarborgen van digitale soevereiniteit. De focus ligt hierbij specifiek op het laten ontwikkelen van software op maat. De inkoop van dienstverlening rondom open source, of de aanschaf van bestaande standaardpakketten, zijn andere disciplines die in toekomstige documenten aan bod zullen komen.

Binnen aanbesteding spelen een veelheid aan niet-functionele eisen een rol. Daarbinnen kan je grofweg een onderscheid maken tussen harde compliance eisen en waardegedreven eisen. Bij harde compliance eisen voldoe je duidelijk wel of niet. Denk daarbij aan open standaarden of informatiebeveiliging. Bij open source gaat het net als bij social return of duurzaamheid om waardegedreven eisen. Er is geen duidelijk normenkader waaraan je moet voldoen zoals de Baseline Informatiebeveiliging Overheid of de verschillende verplichtende lijsten van Forum Standaardisatie dat wel zijn voor hun eigen onderdeel. Wet- en regelgeving geeft wel een inspanningsverplichtingen voor de toepassing van open source als het middel om de transparantie en herbruikbaarheid van digitale technologieën mee te vergroten. Open source kan alleen aan meer doelen bijdragen, zoals leveranciersonafhankelijkheid, soevereiniteit, navolgbaarheid, verkleinen van marktmacht.

Er is dus geen eenduidige set aan eisen neer te leggen waaraan alle aanbestedingen moeten voldoen. Bij open source vraagt dit om bewuste overweging langs de verschillende doelen die je met open source in je aanbesteding zou willen bereiken en in welke mate. Uit die maat blijkt de uiteindelijk ambitie. In dit document zijn die doelen uitgesplitst naar de 4 voordelen van opensourcewerken.

Dat neemt overigens niet weg dat je het hier beschreven laagste ambitieniveau kan zien als een minimale norm. De minimale eisen om te voldoen aan de inspanningsverplichting uit de Wet open overheid en de Wet hergebruik overheidsinformatie.

### 🔍 Gebruik van Generatieve AI

Bij dit document is op drie manieren gebruikgemaakt van Generatieve AI (GenAI):

- GenAI heeft de teksten gereviewd op flow, taal en spelling.
- GenAI is gevraagd aanvullende suggesties te doen op alle teksten.
- GenAI is op specifieke punten gevraagd om te sparren.

Voordat GenAI om input wordt gevraagd, worden alle teksten eerst door de hoofdauteur of bijdragers zelf geschreven. Zo wordt voorkomen dat hier informatie wordt gedeeld waarvan de (hoofd)auteur(s) zelf geen achtergrondkennis heeft/hebben en waarvan onduidelijk is wat de doorwerking is. Door zelf eerst teksten te schrijven, wordt het ook voor GenAI duidelijker wat precies de bedoeling is van een specifiek stuk tekst.

Verder wordt aan dit document in openbaarheid gewerkt. De hier benodigde kennis rond open source, wetgeving en overheidsbeleid is doorgaans niet vertrouwelijk van aard en vrij te gebruiken.

Als er met GenAI is gespard over specifieke teksten, dan is daar in de Rationale een toelichting bij gegeven.

Hierdoor wordt het gebruik van GenAI verantwoord geacht en als waardevolle aanvulling gezien om tot een zo bruikbaar mogelijk instrument te komen.

### 💡 Aanbesteding, opdracht of vacature?

Ondanks dat de teksten zijn geschreven met een aanbesteding in het achterhoofd kunnen ze ook gelezen worden als eisen die je stelt aan inhuurkrachten, in vacatures of je ontwikkelteam. Zo wordt de kennis van git, SBOM-standaarden, open source licenties in deze ambitieladder sowieso veronderstelt. Partijen of personen die in staat zijn een hoger ambitieniveau te bedienen kunnen dan de voorkeur genieten boven andere partijen of personen.

> 💡 **Eigen menu samenstellen?**
>
> In dit document worden suggesties gedaan die je kunt hergebruiken in een aanbesteding. Het zijn expliciet suggesties en geen wetmatigheden. Voel de vrijheid om selectief in deze suggesties te shoppen door bepaalde suggesties te negeren, anders te formuleren, ze anders te combineren of wat voor jouw specifieke situatie het beste werkt.

> 💡 **Gratis of vrij?**
>
> Free as in free speech, not as in free beer - Richard Stallman
>
> Net zoals open source software niet staat voor gratis software, maar voor vrije software is het nastreven van een hoger ambitieniveau niet gratis. Het nastreven van een hoger ambitieniveau heeft consequenties voor jou als aanbestedende dienst. Het vraagt normaliter meer van de eigen organisatie waar het gaat om bijv. de regierol, benodigde financiën, kennis en expertise van de medewerkers, maar ook van de opdrachtnemer(s). Het is ook niet uit te sluiten dat de vijver waaruit je kan vissen bij een hoger ambitieniveau kleiner is, maar je krijgt er wel meer vrijheid voor terug.

## Voordelen van opensourcewerken

Dit document focust zich op de vier voordelen van open source zoals gecommuniceerd door het programma Opensourcewerken. Dat wil niet zeggen dat elk voordeel voor elke situatie even belangrijk is of op dezelfde manier nagestreefd zou moeten worden. Per voordeel zijn dus ambities of opties te formuleren. Deze kan je vrij combineren om je eigen variant samen te stellen. Het is wel aan te raden om zo volledig mogelijk te zijn en dus aandacht aan alle voordelen te besteden. Zo voorkom je dat zo min mogelijk ter interpretatie open blijft. Al is het maar om kenbaar te maken dat er op een bepaald voordeel lage verwachtingen zijn.

1. **Efficiëntie en Onafhankelijkheid**
   Willen we eventueel meerdere leveranciers aanbesteden die ieder een onderdeel van de aanbesteding voor hun rekening nemen? Indien wordt gekozen voor een samenwerking met meerdere opdrachtnemers: hoe wordt regie gevoerd op de samenwerking?
2. **Veiligheid en Betrouwbaarheid**

In hoeverre kunnen we ervanuit gaan dat het ontwikkelde werk veilig is en doet wat het moet doen? Op het gebied van deze eisen worden er in deze ambitieladder geen concessies gedaan. Deze categorie is daarom ook niet in ambities onderverdeeld.

3. **Transparantie en Vertrouwen**
   In hoeverre willen we dat de producten in deze aanbesteding zo open en transparant mogelijk worden ontwikkeld? Of in hoeverre willen we dat in te huren experts met hun kennis en ervaring uitvoering kunnen geven aan een ambitie?
4. **Samenwerking en Innovatie**
   In hoeverre willen we dat leveranciers, maatschappelijke partners (overheid, kennisinstellingen en/of burgers) en/of andere geïnteresseerden tijdens of na de ontwikkeling van de producten (kunnen blijven) samenwerken?

In zijn algemeenheid is het goed om na te denken over hoe de doelstellingen eruit zien op korte, middellange en lange termijn? Welke samenwerking met stakeholders (leveranciers, partners en/of overige geïnteresseerden) is daarvoor nodig over 1, 5 en 10 jaar?

## Voordelen van open source aanbesteden

Open source aanbesteden brengt een groot aantal voordelen met zich mee. Niet alleen voor de opdrachtgever, maar zeker ook voor de opdrachtnemer.

**Voor publieke opdrachtgevers**

- Overheden voldoen ermee aan de **wettelijke inspanningsverplichting** uit de Wet open overheid en de Wet hergebruik overheidsinformatie.
- Overheden voldoen ermee aan de **beleidslijn Open, tenzij**
- Vanaf begin in openbaarheid open source laten ontwikkelen van broncode **verlegt de verantwoordelijkheid** voor een verantwoorde uitvoering hiervan bij de opdrachtnemer. Je mag van een opdrachtnemer verwachten dat zij ook de expertise hebben dat goed te doen.
- Het voorkomt aanzienlijk **reparatie werk achteraf** wanneer de broncode niet transparant-by-design is ontwikkeld.
- Het **voorkomt leveranciersafhankelijkheid** (vendor lock-in) in maatwerk aanbesteding. Door de kennisachterstand van andere leveranciers hebben die doorgaans geen zin om de doorontwikkeling van het werk van een ander op zich te nemen. Wanneer goed

gedocumenteerd en voor hergebruik geschikt in openbaarheid open source wordt ontwikkeld wordt die kennisachterstand geminimaliseerd.
- Opdrachtgever blijven niet **zitten op het eigenaarschap van software**. Het vertegenwoordigd kapitaal die zou kunnen renderen wanneer openbaar open source gemaakt.
- Opdrachtnemers hebben er baat bij met **hergebruik in het achterhoofd** te ontwikkelen.

**Voor opdrachtnemers**

- Opdrachtnemers hebben de **vrijheid het werk** dat ze op kosten van de opdrachtnemer hebben ontwikkeld **in andere (commerciële) contexten her te gebruiken**.
- Opdrachtnemers die een aanbesteding niet winnen, kunnen toch gebruik maken van het werk en de kennis die vanuit de opdracht wordt ontwikkeld. Ze krijgen daarmee maar een **minimale kennisachterstand**. Dat is met name van belang wanneer meerdere leveranciers samen in een specifiek stelsel opereren.
- Het is voor opdrachtnemers makkelijker om een **portfolio** aan te leggen van niet alleen eindproducten, maar ook **met inzicht in de kwaliteit van de techniek**.

Hoe hoger het open source ambitieniveau binnen een aanbesteding hoe groter de kans dat bovenstaande voordelen worden gehaald. Zo zorgt een publieke software aanbesteding niet alleen voor een praktische IT-oplossing voor de opdrachtgever in kwestie, maar leidt de inkoop van nieuwe software ook tot beleidsmatige, publieke, financiële en zelfs commerciële meerwaarde.

## Tekstsuggesties

Per ambitieniveau staan tekst suggesties die gebruikt kunnen worden in de Programma van Eisen van een aanbesteding. Per ambitieniveau veranderen ook de eisen in verwoording, komen er eisen bij of vallen er eisen af. Hoe de eisen verschillen tussen ambitieniveaus is met rood (vervalt) en groen (komt erbij) inzichtelijk gemaakt.

Elk ambitieniveau verwoord een intentie. Ook als er discussie is over de specifieke eisen dan is altijd terug te grijpen op de intentie. Om daarmee alsnog samen met je leverancier tot afspraken te komen die zorgen dat er aan de intentie wordt voldaan.

## Algemene teksten

Een aanbesteding bestaat uit allerlei samenhangende documenten. Het is van belang om in al die documenten op de juiste manier aandacht te besteden aan de eisen die aan het opensourcewerken worden gesteld. Een voorbeeld zin die in het Selectiedocument gebruikt kan worden:

> De, specifiek voor deze Overeenkomst ontwikkelde software, inclusief achterliggende broncode en documentatie, zonder belemmeringen en kosteloos tijdens en na de opdracht ter beschikking te stellen middels resp. een nog nader te bepalen open source (een door de Open Source Initiative goedgekeurd) en creative commons licentie.

Later kan die zin in het Beschrijvend document verder gespecificeerd worden zodra duidelijk is onder welke licentie die ontwikkelingen moeten gebeuren:

> De, specifiek voor deze Overeenkomst ontwikkelde software, inclusief achterliggende broncode en documentatie, zonder belemmeringen en kosteloos tijdens de opdracht ter beschikking stellen middels resp. de European Union Public License (EUPL) v1.2 of hoger en de Creative Common Attribution Share Alike (CC BY-SA).

Alle eisen die onder categorie 1, 2 en 3 staan beschreven zijn bedoeld om op te nemen in de Programma van Eisen. Het is goed om die eisen in samenhang in te leiden met een introducerende zin.

> Bij de ontwikkeling van de prestatie verwacht opdrachtgever dat [enige mate|een hoge standaard|hoogste standaard] van opensourcewerken wordt toegepast. Dat wil zeggen dat:

### 1. Efficiëntie en Onafhankelijkheid

Voor dit voordeel kan je minder spreken over ambities. Het gaat eerder over opties. Voor dit voordeel is het nodig om de aanbesteding vanaf begin af aan op een bepaalde manier in de markt te zetten. Zodat je al tijdens de marktconsultatie aangeeft of je de aanbesteding wel of niet ‘verkavelt’; een chique woord voor opknippen. Dat je bijv. een de (door)ontwikkeling en/of integratie en/of technische/functioneel beheer, hosting etc. allemaal bij dezelfde leverancier wil beleggen of juist bewust van verschillende leveranciers wil afnemen.

**Optie 1. Er wordt één opdrachtnemer aanbesteed om alle dienstverlening te leveren**

Er worden geen aanvullende eisen gesteld aan de aanbesteding. Er wordt één opdrachtnemer gezocht voor alle dienstverlening.

**Optie 2. Er worden meerdere opdrachtnemers aanbesteed om verschillende onderdelen uit de dienstverlening op zich te nemen**

De aanbesteding wordt bewust verkaveld. Er is nagedacht uit welke (onafhankelijke) onderdelen deze aanbesteding bestaat. Het wordt aan potentiële opdrachtnemers duidelijk gemaakt op welke onderdelen ze zich mogen inschrijven. Ook wordt duidelijk gemaakt of opdrachtnemers meerdere onderdelen in de aanbesteding mogen uitvoeren.

**Optie 3. Leveranciers, maatschappelijke partners en/of andere geïnteresseerden werken samen en kunnen blijven samenwerken. Ook nieuwe partners moeten in staat worden gesteld om zich makkelijk bij de samenwerking aan te sluiten.**

Voor de aanbesteding wordt door (één van de) leveranciers of door de opdrachtgevers een gezamenlijke backlog bijgehouden waaraan opdrachtnemende partijen zich commiteren. De uitvoering gebeurt niet op basis van op voorhand volledig uitgedachte werkzaamheden, maar op basis van globale functionele wensen.

## 2. Veiligheid en Betrouwbaarheid

Zoals eerder vermeld wordt er in dit voordeel geen concessies gedaan. Maak gebruik van onderstaande teksten om op een veilige en betrouwbare manier open source software in te kopen.

1. De licenties en de auteursrechthebbende helder worden gecommuniceerd door de gehele broncode en documentatie zoals daarvoor gangbaar is.
2. Er wordt zoveel als mogelijk gebruik gemaakt van bestaande en beproefde open source componenten.
    1. Uit welke componenten de ontwikkelde software bestaat, wordt in elke versie inzichtelijk gemaakt middels een volledige ingevulde SBOM volgens de CycloneDX of de SPDX standaard.

2. Alle bevindingen en/of verbeteringen op bestaande open source componenten dienen te worden gemeld en/of teruggegeven aan de betreffende communities (als upstream verbeteringen).
3. In geval van kwetsbaarheden handelt opdrachtnemer zoals vanuit Coordinated Vulnerability Disclosure beleid van opdrachtgever verwacht mag worden.
4. Bij het gebruik van deze componenten dient de opdrachtnemer ervoor te zorgen dat er geen licentieconflicten ontstaan en dat de voorwaarden waarop ze beschikbaar zijn gesteld correct worden nageleefd.

3. Opdrachtgever verwacht van opdrachtnemer dat hij zich ervan bewust is dat hij bij het gebruik van open source componenten ook een ketenverantwoordelijkheid op zich neemt. Dat wil zeggen dat hij samen met de community zorg draagt voor de veiligheid van de gebruikte componenten.

    1. Mochten er kwetsbaarheden boven tafel komen, dan dient opdrachtnemer direct passende maatregelen te nemen - met een minimale impact op de gebruikers van de dienst - om mogelijk misbruik te voorkomen.
    2. Wanneer een kwetsbaarheid zich voordoet, dan dienen belanghebbenden via passende kanalen direct op de hoogte te worden gesteld waaronder in ieder geval de opdrachtgever.
    3. De kwetsbaarheid dient zo spoedig mogelijk verholpen te worden. Dat kan bijvoorbeeld door het doorvoeren van patches die door de achterliggende community al beschikbaar zijn gesteld, door zelf zorg te dragen voor het (laten) ontwikkelen van een patch die de kwetsbaarheid oplost of door het betreffende component te vervangen door een component met vergelijkbare functionaliteit.

4. Elke versie van de broncode een metadata beschrijving bevat volgens de publiccode.yml standaard.
5. Er modulair wordt gewerkt en modules ook zo veel mogelijk worden ontwikkeld om losstaand (her)gebruik te faciliteren.
6. Gevoelige en geheime informatie niet openbaar wordt gedeeld. De oplossing wordt zo ontwikkeld dat de impact hiervan minimaal is. Denk aan het gebruik van onafhankelijke configuratiebestanden waardoor de broncode zelf wel gepubliceerd kan worden.

    1. Alle gevoelige informatie moet door de opdrachtnemer actief worden bewaakt.

7. Houd de broncode van de applicatie strikt gescheiden van operationele en authenticatiegegevens.

8. Voordat gepubliceerde werken of repositories uit openbaarheid worden gehaald, moet de opdrachtnemer de opdrachtgever een zo volledig mogelijk archief van de repository en de bijbehorende metadata verstrekken, zodat de opdrachtgever deze op zijn eigen platform kan veiligstellen.

## 3. Transparantie en Vertrouwen

**Ambitie 1. Het eenmalig open source publiceren van alle broncode na afronden van de aanbesteding**

1. Alle onder de overeenkomst ontwikkelde broncode en documentatie, onder resp. de [open source licentie] en [documentatie licentie], moeten na afronding van de opdracht op een openbaar git ondersteunend platform worden gedeeld.

**Ambitie 2. Op vaste momenten open source publiceren van de broncode**

1. Alle onder de overeenkomst ontwikkelde broncode en documentatie op afgesproken intervallen of op sleutelmomenten, onder resp. de [open source licentie] en [documentatie licentie], moeten na afronding van de opdracht worden gepubliceerd op een openbaar git ondersteunend platform.
2. Opdrachtnemer het volledige beheer van de software repositories voor zijn rekening neemt.
3. In de documentatie wordt duidelijk gemaakt wat de verschillen zijn tussen twee gepubliceerde versies.

**Ambitie 3. Het volledig openbaar open source ontwikkeling van de broncode**

1. Alle onder de overeenkomst ontwikkelde broncode en documentatie op afgesproken intervallen of op sleutelmomenten in openbaarheid, onder resp. de [open source licentie] en [documentatie licentie], moeten worden ontwikkeld op een openbaar git ondersteunend platform.
2. Opdrachtnemer het volledige beheer van de software repositories voor zijn rekening neemt.
3. In de documentatie wordt duidelijk gemaakt wat de verschillen zijn tussen twee gepubliceerde versies.

4. Zowel de historie en de voortgang van de ontwikkeling volledig te volgen is inclusief de ontwerpkeuzes die tijdens de ontwikkeling zijn gemaakt.

## 4. Samenwerking en Innovatie

### Ambitie 1. Externe bijdragen worden niet behandeld

1. Gepubliceerde versies zijn van elkaar te onderscheiden door consequente versienummering.

### Ambitie 2. Externe bijdragen worden behandeld maar niet actief gezocht

1. Gepubliceerde versies zijn van elkaar te onderscheiden door consequente versienummering.
2. De conventies waaronder code-style, versie nummering, git workflow e.d. inzichtelijk zijn gemaakt.
3. Er goed gedocumenteerd is hoe andere geïnteresseerden een bijdrage kunnen doen aan of vragen kunnen stellen over de broncode en/of documentatie.
4. Alle bijdragen vereisen een ondertekende Contributor License Agreement (CLA) of een geaccordeerde Developer Certificate of Origin (DCO) voordat ze worden geaccepteerd.

**! Contributors License Agreement**

Door sommige organisaties wordt een CLA verplicht gesteld voor alle in openbaar open source ontwikkelde broncode. Mocht dat in jouw organisatie het geval zijn, dan is onderstaande zin te gebruiken:

Er een Contributors License Agreement wordt opgesteld waarbij er aantoonbaar geen bijdragen worden geaccepteerd zonder dat de achterliggende rechtspersoon de CLA heeft ondertekend.

Binnen het Ministerie van Volksgezondheid, Welzijn en Sport wordt de CLA verplicht gesteld. Binnen het Ministerie van Binnenlandse Zaken en Koninkrijksrelaties en het Ministerie van Volkshuisvesting en Ruimtelijke Ordening wordt de toepassing van de CLA juist afgeraden.

**Ambitie 3. Actief samenwerken**

1. De conventies waaronder code-style, versie nummering, git workflow e.d. inzichtelijk zijn gemaakt.
2. Er goed gedocumenteerd is hoe andere geïnteresseerden een bijdrage kunnen doen aan of vragen kunnen stellen over de broncode en/of documentatie.
3. Middels een goed documenteerde governance, conventies, projectdoelstellingen en communicatiekanalen duidelijk wordt gemaakt hoe andere geïnteresseerden kunnen participeren in de samenwerking.
4. Met open documentatie over de werking van de software en ontwerpkeuzes van de achterliggende broncode wordt onmiddellijk hergebruik of toekomstige doorontwikkeling zo optimaal mogelijk gefaciliteerd.
5. Alle bijdragen vereisen een ondertekende Contributor License Agreement (CLA) of een geaccordeerde Developer Certificate of Origin (DCO) voordat ze worden geaccepteerd.

## Voorbeelden

### Laagste ambitie

Oftewel, de basis van optie 1 Efficiëntie en Onafhankelijk + Veiligheid en Betrouwbaar + ambitie 1 van Transparantie en Vertrouwen + ambitie 1 van Samenwerken en Innovatie.

Suggestie voor in het **Selectiedocument** en/of in het **Beschrijvend document**:

> De aanbestedende dienst zoekt één leverancier en verwacht van die leverancier dat alle broncode eenmalig open source zal worden na afronden van de aanbesteding waarbij duidelijk wordt gemaakt dat externe bijdragen niet worden behandeld.

Voor in de **Programma van Eisen**:

> Bij de ontwikkeling van de prestatie verwacht opdrachtgever dat een zekere mate van opensourcewerken wordt toegepast. Dat wil zeggen dat:
>
> 1. Alle onder de overeenkomst ontwikkelde broncode en documentatie, onder resp. [open source licentie] en [documentatie licentie], moeten worden na afronding van de opdracht op een openbaar git ondersteunend platform worden gedeeld.

2. Gepubliceerde versies zijn van elkaar te onderscheiden door consequente versienummering.
3. De licenties en de auteursrechthebbende helder worden gecommuniceerd door de gehele broncode en documentatie zoals daarvoor gangbaar is.
4. Er wordt zoveel als mogelijk gebruik gemaakt van bestaande en beproefde open source componenten.
    1. Uit welke componenten de ontwikkelde software bestaat, wordt in elke versie inzichtelijk gemaakt middels een volledige ingevulde SBOM volgens de CycloneDX of de SPDX standaard.
    2. Alle bevindingen en/of verbeteringen op bestaande open source componenten dienen te worden gemeld en/of teruggegeven aan de betreffende communities.
    3. In geval van kwetsbaarheden handelt opdrachtnemer zoals vanuit Coordinated Vulnerability Disclosure beleid van opdrachtgever verwacht mag worden.
    4. Bij het gebruik van deze componenten dient de opdrachtgever ervoor te zorgen dat er geen licentieconflicten ontstaan en dat de voorwaarden waarop ze beschikbaar zijn gesteld correct worden nageleefd.
5. Opdrachtgever verwacht van opdrachtnemer dat hij zich ervan bewust is dat hij bij het gebruik van open source componenten ook een ketenverantwoordelijkheid op zich neemt. Dat wil zeggen dat hij samen met de community zorg draagt voor de veiligheid van de gebruikte componenten.
    1. Mochten er kwetsbaarheden boven tafel komen, dan dient opdrachtnemer direct passende maatregelen te nemen - met een minimale impact op de gebruikers van de dienst - om mogelijk misbruik te voorkomen.
    2. Wanneer een kwetsbaarheid zich voordoet, dan dienen belanghebbenden via passende kanalen direct op de hoogte te worden gesteld waaronder in ieder geval de opdrachtgever.
    3. De kwetsbaarheid dient zo spoedig mogelijk verholpen te worden. Dat kan bijvoorbeeld door het doorvoeren van patches die door de achterliggende community al beschikbaar zijn gesteld, door zelf zorg te dragen voor het (laten) ontwikkelen van een patch die de kwetsbaarheid oplost of door het betreffende component te vervangen door een component met vergelijkbare functionaliteit.
6. Elke versie van de broncode een metadata beschrijving bevat volgens de publiccode.yml standaard.
7. Er modulair wordt gewerkt en modules ook zo veel mogelijk worden ontwikkeld om losstaand (her)gebruik te faciliteren.

8. Gevoelige en geheime informatie niet openbaar wordt gedeeld. De oplossing wordt zo ontwikkeld dat de impact hiervan minimaal is. Denk aan het gebruik van onafhankelijke configuratiebestanden.
    1. Om welke gevoelige informatie het gaat wordt inzichtelijk gemaakt en actief bijgehouden.
9. Houd de broncode van de applicatie strikt gescheiden van operationele en authenticatiegegevens.
10. Mochten er voornemens zijn de gepubliceerde werken of achterliggende repositories uit de openbaarheid te onttrekken, dan moet de opdrachtgever in de gelegenheid worden gesteld om deze zo volledige mogelijk veilig te stellen op een eigen platform.

### Hoogste ambitie

Oftewel, de basis van optie 3 Efficiëntie en Onafhankelijk + Veiligheid en Betrouwbaar + ambitie 3 van Transparantie en Vertrouwen + ambitie 3 van Samenwerken en Innovatie.

Suggestie voor in het **Selectiedocument** en/of in het **Beschrijvend document**:

> De aanbestedende dienst wil bereiken dat leveranciers, maatschappelijke partners en/of andere geïnteresseerden samen (kunnen blijven) werken. Er wordt dus verwacht dat de broncode volledig openbaar open source wordt ontwikkelt waarbij met open documentatie onmiddellijk hergebruik of toekomstige doorontwikkeling zo optimaal mogelijk wordt gefaciliteerd. Ook nieuwe partners moeten in staat worden gesteld om zich makkelijk bij de samenwerking aan te sluiten.

Voor in de **Programma van Eisen**:

> Bij de ontwikkeling van de prestatie verwacht opdrachtgever dat hoogste standaard van opensourcewerken wordt toegepast. Dat wil zeggen dat:
>
> 1. Alle onder de overeenkomst ontwikkelde broncode en documentatie in openbaarheid, onder resp. de `[open source licentie]` en `[documentatie licentie]`, moeten worden ontwikkeld op een openbaar git ondersteunend platform.
> 2. Opdrachtnemer het volledige beheer van de software repositories voor zijn rekening neemt.
> 3. Zowel de historie en de voortgang van de ontwikkeling volledig te volgen is inclusief de ontwerpkeuzes die tijdens de ontwikkeling zijn gemaakt.

4. De conventies waaronder code-style, versie nummering, git workflow e.d. inzichtelijk zijn gemaakt.
5. Middels een goed documenteerde governance, conventies, projectdoelstellingen en communicatiekanalen duidelijk wordt gemaakt hoe andere geïnteresseerden kunnen participeren in de samenwerking.
6. Met open documentatie over de werking van de software en ontwerpkeuzes van de achterliggende broncode wordt onmiddellijk hergebruik of toekomstige doorontwikkeling zo optimaal mogelijk gefaciliteerd.
7. Alle bijdragen vereisen een ondertekende Contributor License Agreement (CLA) of een geaccordeerde Developer Certificate of Origin (DCO) voordat ze worden geaccepteerd.
8. De licenties en de auteursrechthebbende helder worden gecommuniceerd door de gehele broncode en documentatie zoals daarvoor gangbaar is.
9. Er wordt zoveel als mogelijk gebruik gemaakt van bestaande en beproefde open source componenten.
    1. Uit welke componenten de ontwikkelde software bestaat, wordt in elke versie inzichtelijk gemaakt middels een volledige ingevulde SBOM volgens de CycloneDX of de SPDX standaard.
    2. Alle bevindingen en/of verbeteringen op bestaande open source componenten dienen te worden gemeld en/of teruggegeven aan de betreffende communities.
    3. In geval van kwetsbaarheden handelt opdrachtnemer zoals vanuit Coordinated Vulnerability Disclosure beleid van opdrachtgever verwacht mag worden.
    4. Bij het gebruik van deze componenten dient de opdrachtgever ervoor te zorgen dat er geen licentieconflicten ontstaan en dat de voorwaarden waarop ze beschikbaar zijn gesteld correct worden nageleefd.
10. Opdrachtgever verwacht van opdrachtnemer dat hij zich ervan bewust is dat hij bij het gebruik van open source componenten ook een ketenverantwoordelijkheid op zich neemt. Dat wil zeggen dat hij samen met de community zorg draagt voor de veiligheid van de gebruikte componenten.
    1. Mochten er kwetsbaarheden boven tafel komen, dan dient opdrachtnemer direct passende maatregelen te nemen - met een minimale impact op de gebruikers van de dienst - om mogelijk misbruik te voorkomen.
    2. Wanneer een kwetsbaarheid zich voordoet, dan dienen belanghebbenden via passende kanalen direct op de hoogte te worden gesteld waaronder in ieder geval

> de opdrachtgever.
>
> 3. De kwetsbaarheid dient zo spoedig mogelijk verholpen te worden. Dat kan bijvoorbeeld door het doorvoeren van patches die door de achterliggende community al beschikbaar zijn gesteld, door zelf zorg te dragen voor het (laten) ontwikkelen van een patch die de kwetsbaarheid oplost of door het betreffende component te vervangen door een component met vergelijkbare functionaliteit.
>
> 11. Elke versie van de broncode een metadata beschrijving bevat volgens de publiccode.yml standaard.
> 12. Er modulair wordt gewerkt en modules ook zo veel mogelijk worden ontwikkeld om losstaand (her)gebruik te faciliteren.
> 13. Gevoelige en geheime informatie niet openbaar wordt gedeeld. De oplossing wordt zo ontwikkeld dat de impact hiervan minimaal is. Denk aan het gebruik van onafhankelijke configuratiebestanden.
>
>     1. Om welke gevoelige informatie het gaat wordt inzichtelijk gemaakt en actief bijgehouden.
>
> 14. Houd de broncode van de applicatie strikt gescheiden van operationele en authenticatiegegevens.
> 15. Mochten er voornemens zijn de gepubliceerde werken of achterliggende repositories uit de openbaarheid te onttrekken, dan moet de opdrachtgever in de gelegenheid worden gesteld om deze zo volledige mogelijk veilig te stellen op een eigen platform.

## Veel gestelde vragen door opdrachtnemers

Tijdens een aanbestedingstraject krijgen opdrachtnemers doorgaans verschillende gelegenheden tot het stellen van vragen in de vorm van een Nota van Inlichtingen. De meest gestelde vragen staat hier gedeeld inclusief antwoord suggesties.

- **De aanbesteding vraagt om doorontwikkeling. Betekent dit dat we ook bestaande broncode openbaar open source beschikbaar moeten stellen?**
  - In geval van een Permissive licentie?
    Er is bewust gekozen voor een permissive licentie. Dat betekent dat dit in zijn geheel geen invloed heeft op bestaande broncode van opdrachtnemers.
  - In geval van een Weak-copyleft licentie?
    Er is bewust gekozen voor een weak-copyleft licentie. Dat betekent dat dit in zijn geheel geen invloed heeft op bestaande broncode van opdrachtnemers.

  - In geval van een Strong-copyleft licentie?
    Er is bewust gekozen voor een strong-copyleft licentie. Dat betekent inderdaad dat de software waarbinnen de doorontwikkeling plaatsvind onder dezelfde licentie openbaar open source gemaakt dient te worden.

- **Mogen we componenten van derden gebruiken waar wij als opdrachtnemer het intellectueel eigendom niet van hebben?**
  Dat mag zeker en we moedigen het gebruik van beproefde open source componenten zelfs aan. Uit welke componenten de voor deze opdracht ontwikkelde werken zijn opgebouwd zal inzichtelijk moeten worden gemaakt als onderdeel van de gevraagde documentatie. Het is wel de verantwoordelijkheid van de opdrachtnemer om ervoor te zorgen dat er geen licentieconflicten ontstaan onder deze overeenkomst ontwikkelde werk en dat de voorwaarden waarop bestaande open source componenten beschikbaar zijn gesteld worden nageleefd.
- **Zijn we ook verantwoordelijk voor kwetsbaarheden die worden gevonden in componenten van derden?**
  Opdrachtnemer heeft een ketenverantwoordelijkheid om (samen met de community) de kwetsbaarheid zo snel mogelijk op te lossen en de mogelijk gevolgen te minimaliseren. Verdere details staan in de Programma van Eisen onder het thema open source. Welke specifieke maatregelen er worden verwacht en welke normen er moeten worden gevolgd om de veiligheid van de broncode op voorhand zoveel mogelijk te garanderen staat verder gespecificeerd in de Programma van Eisen onder het thema informatiebeveiliging.
- **Waar ligt het intellectueel eigendom van de opgeleverde broncode?**
  Op deze aanbesteding zijn de [algemene aanbestedingsvoorwaarden] van toepassing. Deze bepalen dat de intellectuele eigendomsrechten van al het maatwerk dat onder deze overeenkomst wordt ontwikkeld, toebehoort aan de opdrachtgever. Dit omvat het volledige recht voor iedereen om de broncode te gebruiken, te wijzigen en te distribueren (inclusief het openbaar maken ervan als open source), zonder verdere toestemming of vergoeding aan de opdrachtnemer.
- **Mag iedereen met het open source ontwikkelde werk doen wat die wil?**
  Opdrachtgever laat het een ieder vrij om de uit deze overeenkomst ontwikkelde werken her te gebruiken in welke (commerciële) context dan ook, zoals ook gangbaar is bij open source ontwikkeling. Voor het hergebruik binnen de context van deze aanbesteding gelden natuurlijk de daarvoor geldende waarborgen.
- **Hoe verwacht opdrachtgever dat opdrachtnemers geld kunnen verdienen aan open source code?**

Opdrachtnemer krijgen betaald voor alle ontwikkeling die in het kader van deze aanbesteding worden gevraagd. Het intellectueel eigendom van de werken die in opdracht van deze aanbesteding worden ontwikkeld liggen bij de opdrachtgever. Het staat opdrachtnemer natuurlijk vrij om — gezien het open source karakter van deze aanbesteding — de ontwikkelde broncode her te gebruiken in welke andere (commerciële) context dan ook.

- **Welke eisen worden er gesteld aan de manier waarop we in de openbaarheid open source moeten werken?**
  Het staat opdrachtnemer vrij om keuzes te maken te passen bij de eigen werkwijze. Zolang er maar voldaan wordt aan wat er in de Programma van Eisen wordt gevraagd.
- **Kan opdrachtgever aangeven welke beveiligingseisen er worden gesteld aan de repository waarin alle open source ontwikkeling dient plaats te vinden?**
  Wat betreft de beveiliging van de openbare repository verwachten we minimaal:
  - Toegang tot de repository loopt via functioneel beheer van opdrachtgever.
  - Alle commits moeten geverifieerd gesigned worden.
  - Wijzigingen gebeuren alleen via Pull Requests met twee approvals.
  - Schakel codescanning, scanning naar secrets en afhankelijkheden in en onderhoud deze (met bijv. GitHub Advanced Security of vergelijkbare programmatuur).
- **Welke risico's zien jullie zelf in de gekozen aanpak van publieke open source ontwikkeling, en hoe zijn deze tot nu toe gemitigeerd?**
  We zien daarbij twee soorten risico's: beveiligingsrisico en imago-risico. Beveiligingsrisico in de zin dat een publieke repository een extra informatiebron is voor kwaadwillenden. Wij vinden echter de beveiligingsrisico's van closed source groter dan die van open source, omdat kwetsbaarheden langer onder de radar kunnen blijven. Imago-risico in de zin dat ondermaats werk de reputatie van de opdrachtgever en daarmee het vertrouwen in het gebruik van het product schaadt.

  Beide risico's worden gemitigeerd, door het aantal dependencies te beperken, door technische keuzes te documenteren en door mensen uit de community uit te nodigen feedback te geven en door professioneel en secuur te werken.
- **Zijn er specifieke eisen of richtlijnen met betrekking tot het beheren van bijdragen van derden (bijvoorbeeld pull requests) aan de openbare repository?**
  Er moet een (minimaal) proces worden ingericht. Voornaamste onderdeel van dat proces is dat externe partijen een CLA (Contributor License Agreement) moeten ondertekenen.

Voor het beoordelen en goedkeuren van de wijzigingen is de aanname dat hiervoor hetzelfde proces gevolgd wordt als voor bijdrages van binnen het team.

- **Zijn er alternatieven overwogen voor volledige open source publicatie vanaf de start? Zo ja, waarom zijn deze afgewezen?**
  Er is overwogen om pas tot open source publicatie over te gaan na een opstart-fase. Dit is afgewezen, om te voorkomen dat er eerst een opschoningsactie nodig geacht wordt, voordat er tot publicatie over gegaan kan worden.

## Aanbevelingen

In dit hoofdstuk zijn twee type aanbevelingen te vinden. Aanbevelingen die bedoeld zijn als suggestie en diegene die bedoeld zijn als waarschuwing. In de suggesties staan tips die ter overweging meegenomen kunnen worden in de opdracht. In de waarschuwingen staan overwegingen die bedoeld zijn om opdrachtgever bewust te maken van belangrijke overwegingen die zwaarwegende consequenties kunnen hebben.

### 💡 Doorontwikkeling

Alle ambities zijn geschreven vanuit het uitgangspunt van een volledig maatwerk ontwikkelde oplossing of component. Natuurlijk wordt er ook doorontwikkeld op bestaande (open source) oplossingen of componenten. Het verdient de voorkeur om bij doorontwikkeling zoveel mogelijk aansluiting te zoeken bij de achterliggende community. Dat is de snelste manier om nieuwe ontwikkelingen ten goede te laten komen van het algemeen belang.

Mocht dat om redenen niet lukken, denk aan doorlooptijd om wijzigingen verwerkt te krijgen, onwelwillendheid mee te werken of simpelweg gebrek aan kennis en kunde, dan kan doorontwikkeling op bestaande open source ook plaatsvinden op een kopie (ook wel fork genoemd). Zo kan de continuïteit van de opdracht alsnog gewaarborgd blijven en het geeft de opdrachtnemer alle regie op het (door) te ontwikkelen product en bijbehorende repositories. Een klein nadeel van het werken op een kopie is dan weer wel dat de opdrachtnemer (mede-)verantwoordelijk wordt voor het onderhoud, de documentatie, veiligheid etc. van die kopie.

Ook bij doorontwikkeling op een kopie is het de bedoeling dat verbeteringen conform de ambitie ten goede komen van de gemeenschap achter het oorspronkelijke werk. Door ze in

ieder geval op de hoogte te stellen van alle ontwikkelingen en medewerking aan te bieden wanneer er (alsnog) bereidheid ontstaat de wijzigingen door te voeren. Zodat idealiter de noodzaak voor de kopie verdwijnt. Hier moet bij de ontwikkeling op die kopie natuurlijk wel rekening mee gehouden worden. Dat er een ingebouwde flexibiliteit is om makkelijk te wisselen tussen versies; kopie of oorspronkelijke (bijgewerkte) oplossing of component.

Wanneer er door wordt ontwikkeld op een closed source product, dan kan die doorontwikkeling beschouwd worden als een losstaande open source ontwikkeling.

### 💡 Sponsoren van open source (componenten)

Open source software kan alleen bestaan door de community die samen aan de software werkt en er ook het beheer op doet. Daarbij kan het gaan om vrijwilligers, maar ook commerciële of non-profit organisaties. Dat neemt niet weg dat je bij het gebruik van open source ook op een of andere manier de morele plicht hebt eraan bij te dragen. Dat kan in natura zoals voorgesteld in de huidige eisen. Wanneer er verbeteringen worden doorgevoerd op bestaande broncode, dan moeten deze terugvloeien naar de achterliggende communities. Je zet dus concrete ontwikkelcapaciteit in ten behoeve van de doorontwikkeling van bestaande code. Een andere manier is via een financiële bijdrage.

Een financiële bijdrage leveren aan een specifiek open source project is alleen een uitdaging. De overheid kent in zijn algemeenheid maar twee manieren om geld uit te geven. Via een aanbesteding of via een subsidie. Het ‘nadeel’ van beide methoden is dat je er niet mee mag discrimineren. Je kan niet bepaalde partijen op voorhand aanspraak laten maken op een aanbesteding of subsidie. Wanneer je dus een subsidieregeling wil starten voor open source, dan maken alle open source projecten daar aanspraak op. Je geld kan dan terecht komen bij ontwikkelaars die helemaal niet werken aan componenten in gebruik in je project. Of je dat als een probleem ervaart is een tweede.

Er is wel een andere manier. Dat is het gebruiken van de opdrachtnemer als een proxy voor de financiële gelden. Dat is wat er nu ook gebeurt bij de social return. Het is gangbaar 2% van de opdrachtwaarde ten gunste te laten komen van social return. Het is dan aan de opdrachtnemer te bepalen waar het geld dan naartoe gaat, zolang het maar doelen zijn die ten gunste komen van de social return eisen. Op dezelfde manier kan je een opdrachtnemer vragen een X percentage van de opdrachtwaarde evenwichtig te doneren aan de open source projecten waar de opdrachtnemer gebruik van maakt. Een andere mogelijkheid is doneren aan fondsen die open source projecten ondersteunen zoals NLnet of de Sovereign Tech Agency.

Stel nu dat je een open source donatie percentage afspreekt van 2%, er € 15 miljoen is gemoeid met de opdracht en er 500 open source componenten worden gebruikt, dan zou elk component € 600,- ontvangen als donatie. Om nu aan 500 verschillende open source projecten te doneren geeft gelijk een aanzienlijke werklast, dus daarin is te variëren. Het is aan de opdrachtgever om creatief te zijn in hoe je deze eis stelt. Een voorbeeld:

> Opdrachtgever verwacht dat opdrachtnemer van de totale waarde van deze opdracht 2% ten goede laat komen aan de open source componenten die opdrachtnemer gebruikt voor de ontwikkeling van de gevraagde prestatie. Het staat opdrachtnemer vrij om dit bedrag naar eigen inzicht te verdelen. De enige aanvullende eis is dat er aan tenminste 10 open source projecten wordt gedoneerd die ondersteund worden door een erkende non-profit stichting of vereniging (in het engels foundation).

In dit geval ontvangen de 10 open source projecten elk € 30.000,-. Voor veel open source projecten een aanzienlijk bedrag.

### ❗ Het beteugelen van strong-copyleft

Wanneer er voor een strong-copyleft open source licentie wordt gekozen is het goed om oog te houden voor het virale effect van deze licentiegroep. Het is wijs de invloed van de licentie beperkt te houden tot wat je zelf kan overzien. Concreet betekent dit dat je probeert te voorkomen dat er onvoorziene effecten ontstaan door de toepassing van dit type licentie. Daarvoor zou deze zin gebruikt kunnen worden:

> In de uitwerking van de verschillende architecturen wordt rekening gehouden met de virale werking van de gekozen sterk wederkerige (strong-copyleft) licentie. Dat wil zeggen dat de architecturen zo worden ontworpen dat deze doorwerking beperkt blijft tot de prestatie die vanuit deze aanbesteding wordt gevraagd en geen invloed heeft op het gebruik van de prestatie of op de diensten die eventueel op de prestatie worden aangesloten.

### ❗ Wel of geen Contributors License Agreement (CLA) of Developer Certificate of Origin (DCO)?

Bij open source projecten komt het vaak voor dat meerdere partijen bijdragen aan dezelfde code. Het is dan handig om van tevoren duidelijk te maken hoe deze bijdrages juridisch worden

geregeld. Dat hoeft niet ingewikkeld te zijn, maar helpt wel om misverstanden te voorkomen over wie wat mag doen met de code en onder welke voorwaarden.

**Contributors License Agreement (CLA)**

Een Contributors License Agreement (CLA) is een manier om afspraken vast te leggen tussen een bijdrager en het project. In de overeenkomst staat dat de bijdrager toestemming heeft om zijn bijdrage te doen, onder de voorwaarden die het ontvangende project stelt. Het intellectueel eigendom van de code blijft altijd bij de bijdrager.

Een CLA vangt over het algemeen twee zaken af: * Het geeft het project de vrijheid om later van licentie te wisselen zonder iedere bijdrager opnieuw om toestemming te hoeven vragen. Zonder CLA zou dat wél moeten, wat bij onbereikbare of weigerende bijdragers kan leiden tot een zogenoemde license lock-in. * Het biedt extra zekerheid tegen onverwachte auteursrechtschendingen. De bijdrager verklaart namelijk dat hij gerechtigd is de bijdrage te leveren en daar de verantwoordelijkheid voor neemt.

Belangrijk: ook zonder CLA blijft de gekozen open source licentie altijd leidend. De CLA is dus een aanvulling, geen vervanging.

**Developer Certificate of Origin (DCO)**

Een andere, lichtere manier om de juridische afspraken vast te leggen is het Developer Certificate of Origin (DCO). Bij een DCO verklaart een bijdrager bij elke bijdrage dat hij het recht heeft om de code bij te dragen en dat hij akkoord gaat met de licentie van het project. Dit gebeurt vaak via een eenvoudige check bij het indienen van een commit of pull-request. Het voordeel is dat het minder formeel is dan een CLA - hij hoeft immers niet ondertekend te worden - maar toch voldoende zekerheid biedt dat de bijdragen correct zijn afgegeven.

**Kiezen tussen CLA en DCO**

Projecten kunnen kiezen voor een CLA, een DCO, beide of geen van beide, afhankelijk van de omvang, complexiteit en juridische gevoeligheid van het project. Wat belangrijk is, is dat de regels duidelijk zijn voor iedereen die bijdraagt. Zo weet een externe bijdrager precies onder welke voorwaarden zijn code wordt geaccepteerd en voorkomt het dat later discussies ontstaan over rechten of gebruik.

**Geen CLA of DCO**

Zelfs zonder CLA of DCO is het dus goed mogelijk om een open source project te beheren. In dat geval geldt automatisch dat bijdragers akkoord gaan met de open source licentie van het

project, maar het kan handig zijn dit expliciet te vermelden in de documentatie of in een pull request template, zodat het voor iedereen duidelijk is.

**Licenties en eigendom**

Het is belangrijk te beseffen dat de projectlicentie niet automatisch geldt voor alle afzonderlijke componenten. Zeker niet wanneer je die componenten ook los van het project wilt hergebruiken. De projectlicentie bepaalt onder welke voorwaarden het geheel – de verzameling van alle onderdelen – wordt gepubliceerd. Zo kan een project in zijn geheel onder de EUPL staan, terwijl sommige losse bestanden of modules nog onder MIT of Apache hergebruikt kunnen worden.

Voor de delen van de code waar je zelf de IE-rechten van bezit, kun je altijd nog van licentie wisselen. Zodra de code echter is vermengd met bijdragen van derden, zit je vast aan de voorwaarden waaronder zij hun bijdrage hebben geleverd. Een licentiewijziging kan dan alleen met hun toestemming. Let er ook op dat een licentiewijziging van een module gevolgen kan hebben voor de licentie van het project als geheel. Zo kan een project als geheel niet langer onder de EUPL blijven, wanneer een losse module van MIT naar GPL wijzigt – althans niet meer in combinatie met die specifieke versie van de module.

## Rationale

Op het moment van schrijven van dit document is het niet mogelijk om commentaar uit HackMD te exporteren. Het is dus niet makkelijk inzichtelijk te maken welke interacties (met wie) tot een bepaalde wijziging hebben geleid. Om die informatie niet verloren te laten gaan zal een samenvatting van die gesprekken, en de eventuele wijzigingen waartoe ze geleid hebben, hier weergegeven worden.

### Ambities vanuit opensource werken

15 januari 2025

In de eerste aanzet van de ambitieladder was het idee om alle eisen te variëren in een oplopende schaal van 1 tot 10. Of het 10 ambities moesten worden stond overigens niet op voorhand vast. Het was vooral bedoeld om met elkaar over na te gaan denken. Ambitie 1 en 8, 9 en 10 waren daarvan al uitgewerkt. Een goede invulling geven op de tussenliggende ambities

bleek moeilijker. Vanuit de community werden de volgende opmerkingen gemaakt over deze opzet:

- Zitten er in de ambities niet impliciet assen verscholen? In deze eerste opzet wordt gevarieerd over de assen transparantie en samenwerking.
- Als je die assen kan expliciteren, dan kan je vervolgens per as ambities maken die samengevoegd tot een set van eisen leiden.
- Is het niet handiger aan te sluiten bij de vier categorieën van opensourcewerken?

We waren het er snel over eens dat het definiëren van teveel assen en teveel ambities niet werkbaar zouden. In de huidige opzet van 3 opties x 3 ambities x 3 ambities x 1 vaste set krijg je al 27 potentieel verschillende varianten. Het aansluiten bij de vier categorieën van opensourcewerken was daarin een compromis. Genoeg variatie zonder teveel verschillende samengestelde varianten te krijgen. En ook, eenduidige communicatie over opensourcewerken vanuit de Rijksoverheid. Op drie van de categorieën zijn nu dus 3 ambities/opties geformuleerd vanuit het idee van de assen. De combinatie van die ambities/opties leiden dan tot een zeker ambitieniveau en daarmee een samenhangende set aan eisen.

### Waarom een Contributors License Agreement (CLA)?

15 januari 2025

In de basisset aan eisen is de volgende eis opgenomen: > Er een Contributors License Agreement aanwezig is waarbij er aantoonbaar geen bijdragen worden geaccepteerd zonder dat de achterliggende rechtspersoon de CLA heeft ondertekend.

Deze eis leidde vanzelfsprekend tot discussie. Een CLA schrikt sommige ontwikkelaars af en er organisaties die als beleidsuitgangspunt hebben dat ze niet ontwikkelen aan projecten waar een CLA wordt gevraagd. De reden om de eis toch op te nemen in de basisset is meerledig:

1. Het gaat hier om een eis aan derden. Aan degene die willen bijdragen aan broncode die uit de opdracht is ontstaan. Het is geen richtsnoer voor de eigen ontwikkelaars in geval zij bijdragen aan projecten van derden.
2. Het goed doorzien van de consequenties van een licentiekeuze vraagt enige mate van expertise. Helemaal in relatie tot het accepteren van bijdragen van derden. De ambitieladder moet toepasbaar zijn voor ICT-generalisten. Better safe than sorry.
3. Het stopzetten van verplichtstelling van de CLA kan wel, terwijl het achteraf invoeren ervan geen zin heeft. Tenminste, niet als er al bijdragen van derden zijn geaccepteerd.

> Zodra een project enige volwassenheid heeft bereikt kan altijd voor het afschaffen van verplichtstelling worden gekozen.

Er is gekozen om onder aanbevelingen extra uitleg te geven over de toepassing van de CLA.

13 september 2025

De CLA heeft in de basis twee doelen: het voorkomen van license lock-in en auteursrechtschendingen. Voor het voorkomen van dat tweede is er echter ook een vriendelijker document, namelijk de Developer Certificate of Origin (DCO). Een DCO hoeft niet officieel ondertekend te worden, maar kan simpelweg geaccordeerd worden. Dat maakt de DCO algemeen breder geaccepteerd door ontwikkelaars of organisaties. De DCO is om die reden aangevuld als alternatief instrument van de CLA met een aanvullende uitleg in de aanbevelingen. De DCO voorkomt echter niet dat je in een license lock-in situatie terecht kan komen.

### Welke mate van veiligheid kan een leverancier garanderen?

2 februari 2025

In de ambitieladder stonden de volgende eisen waar het gaat over veilige code: > 1. Opdrachtnemer draagt zorg voor het opleveren van veilige broncode. > 1. Mochten er kwetsbaarheden boven tafel komen, dan dient opdrachtnemer direct passende maatregelen te nemen - met een minimale impact op de gebruikers van de dienst - om mogelijk misbruik te voorkomen. > 2. Wanneer een kwetsbaarheid zich voordoet, dan dienen belanghebbenden via passende kanalen direct op de hoogte te worden gesteld waaronder in ieder geval de opdrachtgever.

Hierop werd gereageerd dat het onduidelijk is wie verantwoordelijk is voor het daadwerkelijk oplossen van de kwetsbaarheid. Dit werd in de eerste versie teveel geïmpliceerd vanuit de tekst direct passende maatregelen. De feedbackgever stelde terecht de vraag wie het probleem dan op moet lossen en wie er dan voor zou moeten betalen. Stel je jezelf volledig afhankelijk op van de community? Moet de opdrachtgever het zelf oplossen of mag die daarvoor ook iemand inhuren? Waarbij bij inhuren ook aan leden van de community zelf gedacht kan worden. Daarop is de volgende regel toegevoegd:

> 3. De kwetsbaarheid dient zo spoedig mogelijk verholpen te worden. Dat kan door het doorvoeren van patches die door de achterliggende community al beschikbaar zijn gesteld of door zelf zorg te dragen voor het (laten) ontwikkelen van een patch die de

> kwetsbaarheid oplost.

Op deze toevoeging kwam vervolgens weer een aanvulling dat de opdrachtgever ook de mogelijkheid heeft om het betreffende component in zijn geheel te vervangen door een component met vergelijkbare functionaliteiten. Die is vervolgens ook toegevoegd. Waarbij al snel de conclusie was dat een uitputtend lijstje van mogelijkheden nooit gegeven kan worden. De punten onder 3 zijn vervolgens vertaald naar suggesties. Zo is duidelijk aan welke maatregelen opdrachtgever zoal denkt.

Op de hele set aan eisen rond veilige broncode ontstond een abstractere discussie. Namelijk dat je er niet vanuit kan gaan dat het geleverde werk zonder kwetsbaarheden is. Met de tekstsuggesties: > Opdrachtgever en opdrachtnemer zijn er zich van bewust dat er nog onbekende kwetsbaarheden in het geleverde werk aanwezig zullen zijn.

De kern is volgens de feedbackgever dat altijd alles kwetsbaarheden bevat, deze veelal nog onbekend zijn, en het er dus om gaat hoe opdrachtgever en -nemer samen afspraken maken over hoe om te gaan met deze wetenschap. Dat dat niet iets is wat je alleen bij de opdrachtnemer neer kan leggen, maar wat een gezamenlijke inspanning vereist.

Verder gaf de feedbackgever aan dat een respectvolle communicatie ook van belang is. Als software toch kwetsbaarheden blijkt te bevatten, dan moet dat niet leiden tot een schuldvraag. Dat iemand gefaald zou hebben. De schuldvraag komt pas om de hoek wanneer opdrachtnemer zijn (afgesproken) professionele verantwoordelijkheid niet heeft genomen.

13 februari 2025

De uiteindelijke conclusie is dat veiligheid niet iets is wat binnen de scope van open source inkoopeisen valt. Het is aan informatiebeveiliging om specifieker in te gaan op de normen en instrumenten die specifiek gaan over de veiligheid van broncode. Waar open source een aanvulling op doet is het relationele aspect. Dat je bij het gebruik van vrije software componenten ook een plicht hebt om er waar mogelijk eraan bij te dragen. Die bijdrage kan allerlei vormen aannemen. Van het melden van een bug tot het daadwerkelijk aandragen van een oplossing ervoor, een financiële bijdrage doen of daadwerkelijk onderdeel worden van die community. Uiteindelijk is dit vertaald naar de term ketenverantwoordelijkheid.

> 3. Opdrachtgever verwacht van opdrachtnemer dat hij zich ervan bewust is dat hij bij het gebruik van open source componenten ook een ketenverantwoordelijkheid op

> zich neemt. Dat wil zeggen dat hij samen met de community zorg draagt voor de veiligheid van de gebruikte componenten.

**Doorontwikkeling in een fork of niet?**

2 februari 2025

Vanuit de community kwam de vraag of je ambitieladder alleen kunt gebruiken bij een volledige nieuw te ontwikkelen maatwerk of ook bij doorontwikkeling van bestaand (closed source) oplossingen? Daarop is het informatieblokje toegevoegd aan de introductietekst. Daarin staat vermeld dat wanneer doorontwikkeling plaatsvind in een fork dit gezien kan worden als de ontwikkeling van een kant en klaar product. De opdrachtnemer heeft immers zelf de volledige regie op de fork en je maakt je niet afhankelijk van de achterliggende community als het gaat om bedrijfscontinuïteit.

Niet iedereen was het eens met de die logica. Immers, als je doorontwikkeld op een fork, dan is nog maar de vraag of de wijzigingen ook daadwerkelijk terugvloeien naar de achterliggende community. Daarbij werd de volgende tekstsuggestie gedaan: > Het wordt aanbevolen om in geval van doorontwikkeling op bestaande open source dit te laten plaatsvinden als bijdrage op de bestaande codebase. Zo wordt de open source code base en community verder versterkt, door niet alleen te gebruiken wat er al is, maar ook bij te dragen in de bestaande codebase.

Het probleem is alleen dat de achterliggende communities niet altijd wijzigingen aan de bestaande codebase overnemen. Kan dan verschillende redenen hebben:

1. De wijzigingen worden niet als verbetering gezien omdat het bijv. niet past bij de visie van de community of te context specifiek is.
2. De community heeft niet de kennis en kunde in huis om de nieuwe wijzigingen ook te kunnen beheren.
3. De community heeft niet de capaciteit om de verbeteringen te beoordelen in het tempo dat de opdracht verlangt.

Het werken op een fork geeft de meeste zekerheden dat de doorontwikkeling ook daadwerkelijk leidt tot het doel dat de opdrachtgever voor ogen had. In de ambitieladder zelf staat vervolgens de eis: > Alle bevindingen en/of verbeteringen op bestaande open source componenten dienen te worden gemeld en/of teruggegeven aan de betreffende communities.

Er wordt dus wel degelijk verwacht van de opdrachtgever dat deze verbeteringen ook ten gunste van de achterliggende community laat zijn. Maar dan wel los van de vraag wat hoe de

community dit zelf beoordeeld.

Desondanks is het ideaal van opensourcewerken natuurlijk dat je in goede harmonie kan samenwerking aan software, met idealiter de achterliggende community. De oorspronkelijke tekst is aangevuld vanuit dit ideaal, maar de kopie als tweede optie.

## Projecten

Deze ambitieladder is eerder succesvol toegepast in de volgende projecten

| Project | Organisatie | Waarde |
|---|---|---|
| Persoonlijke Gezondheidsomgeving | Ministerie van Volksgezondheid, Welzijn en Sport | € 15,- miljoen |
| CumuluZ subsidie t.b.v. het Landelijk Dekkend Netwerk | Ministerie van Volksgezondheid, Welzijn en Sport | € 11,- miljoen |
| PoC's en Pilots t.b.v. Generieke Functies | Ministerie van Volksgezondheid, Welzijn en Sport | € 10,- miljoen |
| Elektronisch Kandidaatstellingssysteem (e-KS) | De Kiesraad | € 3 tot 5,- miljoen |

## Bijdragers

Dank aan alle hier bij naam genoemd, maar ook alle bijdragers die graag anoniem willen blijven.

- Maurice Hendriks (Hoofdauteur; Ministerie van Volksgezondheid, Welzijn en Sport)
- Marc van Andel (Kadaster)
- Joeri Bekker (Maykin)
- Mike Gifford (CivicActions)

- Johan Groenen (Tiltshift)
- Mitch Hak (Ministerie van Volksgezondheid, Welzijn en Sport)
- Rutger Haagsma (Ritense)
- David Heijkamp (Naturalis)
- Marlena van Ooijen (Logius)
- Nico Rikken (Alliander)
- Eva van Sloten (Ministerie van Binnenlandse Zaken en Koninkrijksrelaties)
- Job Spierings (Amsterdams Fonds voor de Kunst)
- Manfred Zielinski (Ministerie van Binnelandse Zaken en Koninkrijksrelaties)

---