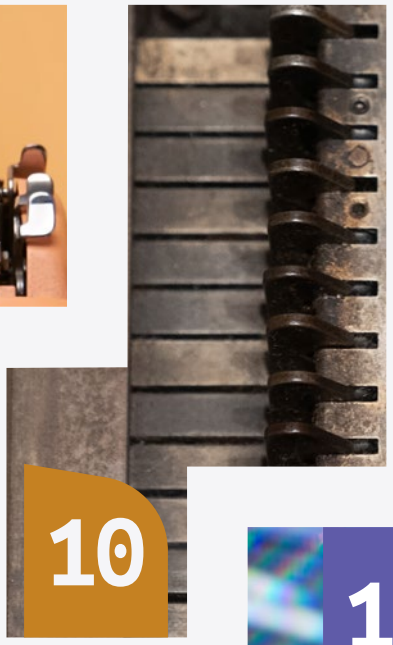
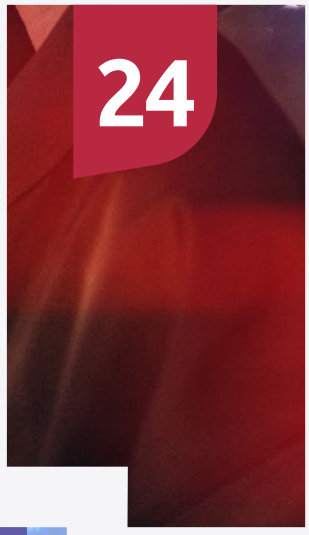
sogeti labs
Part of Capgemini

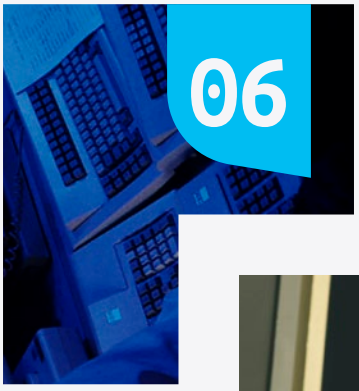
Vibe coding

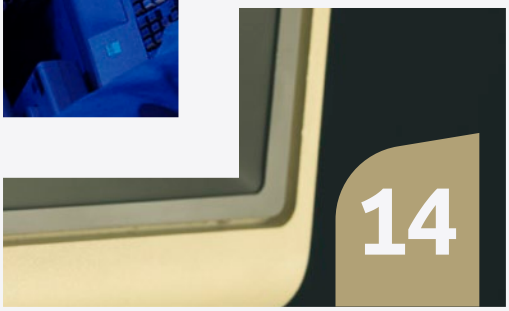
Rapport 2 uit de **Autopilot**-serie

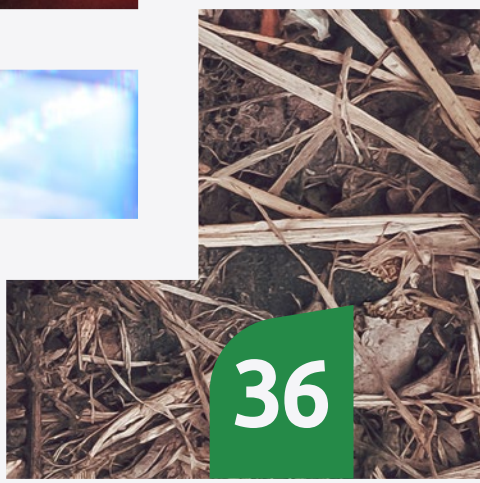
Inhoud

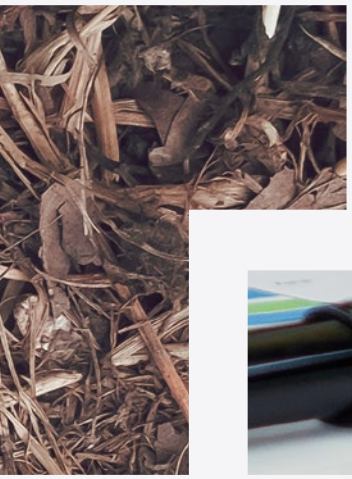


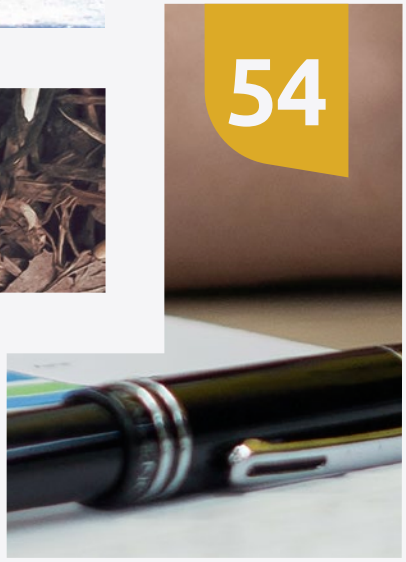


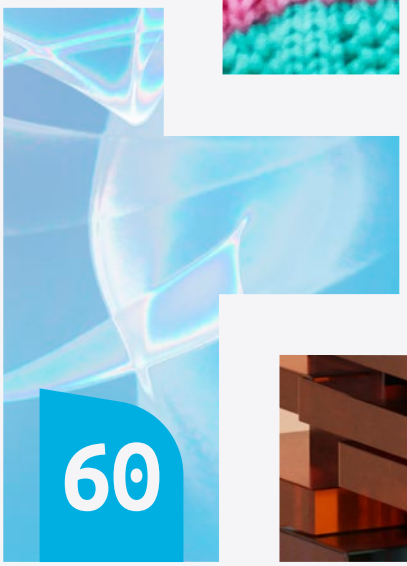


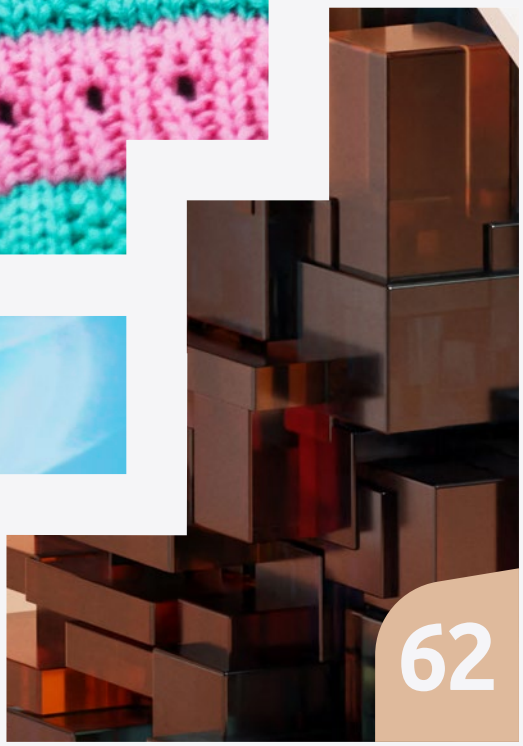




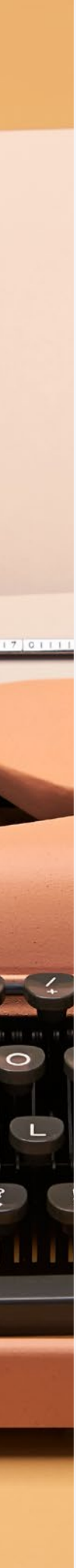








Inleiding: Vibe coding



Er voltrekt zich een stille, maar wezenlijke verschuiving in de manier waarop werk tot stand komt. Niet alleen in de tools die we hanteren, maar in de wijze waarop denken, spreken en bouwen in elkaar overvloeien. De opkomst van *vibe coding* – een AI-gestuurde praktijk waarbij intenties in natuurlijke taal worden omgezet in functionerende ontwerpen of code – maakt zichtbaar hoe technologie de grondvormen van werk opnieuw ordent.

Wat begon als een experimenteel hulpmiddel voor ontwikkelaars, vindt inmiddels ook toepassing in andere domeinen: van marketing en HR tot consultancy en besluitvorming. In sommige gevallen volstaat een enkele prompt om werk te genereren dat eerder door meerdere mensen werd voorbereid of uitgevoerd. Daarmee verschuift niet alleen de uitvoering, maar ook de coördinatie, de rolverdeling en de opvatting van wat 'productief zijn' betekent.

In dit rapport verkennen we hoe deze ontwikkeling zich manifesteert. Dit doen we aan de hand van drie scenario's voor de toekomst van werk. We onderzoeken hoe organisaties zich mogelijk bewegen richting een vorm van *vibe-architectuur*: een infrastructuur waarin AI-systemen niet alleen ondersteunen, maar meedenken, meebouwen en soms zelfs de aanzet geven tot besluiten. We bespreken daarbij zowel concrete praktijkvoorbeelden als de organisatorische en ethische vragen die deze verschuiving oproept.

Voor bestuurders en beleidsmakers is het van belang deze trend niet te interpreteren als tijdelijke hype, maar als een aanwijzing van onderliggende structurele veranderingen. Dit rapport biedt geen blauwdruk, maar wel begrippen, vragen en richtingaanwijzers die kunnen helpen om op die veranderingen te anticiperen.

Misschien is de manier waarop dit rapport tot stand kwam nog het meest illustratief voor de verschuiving die we beschrijven. Het werd niet lineair geschreven, maar in wisselwerking met een taalmodel. Niet achter gesloten deuren, maar in fragmenten: onderweg in de auto, tussen afspraken door, aan de keukentafel. Door vragen te stellen, ideeën te toetsen en hardop te denken in dialoog met AI ontstond een tekst die gevormd werd door *vibe writing,* een schrijfstijl waarin intentie en gegenereerde taal zich geleidelijk tot inhoud verdichten. Wat u leest is dus niet alleen een beschrijving van een mogelijke toekomst, maar een voorproef ervan.

Hoofdstuk 1
De geboorte van software engineering

Toen in 1968 een groep ingenieurs, wetenschappers en militairen zich in Garmisch-Partenkirchen verzamelde, klonk voor het eerst de term *software engineering*. Het was geen bestaande discipline, maar een dringende oproep om er een te scheppen. Men sprak van een softwarecrisis: systemen groeiden sneller dan ons begrip ervan. Grootschalige projecten liepen uit de hand, soms met levensgevaarlijke consequenties, omdat niemand nog kon doorgronden hoe de code werkte. Het moment markeerde een kantelpunt: software moest voortaan worden ontworpen met dezelfde ernst als bruggen of vliegtuigen.

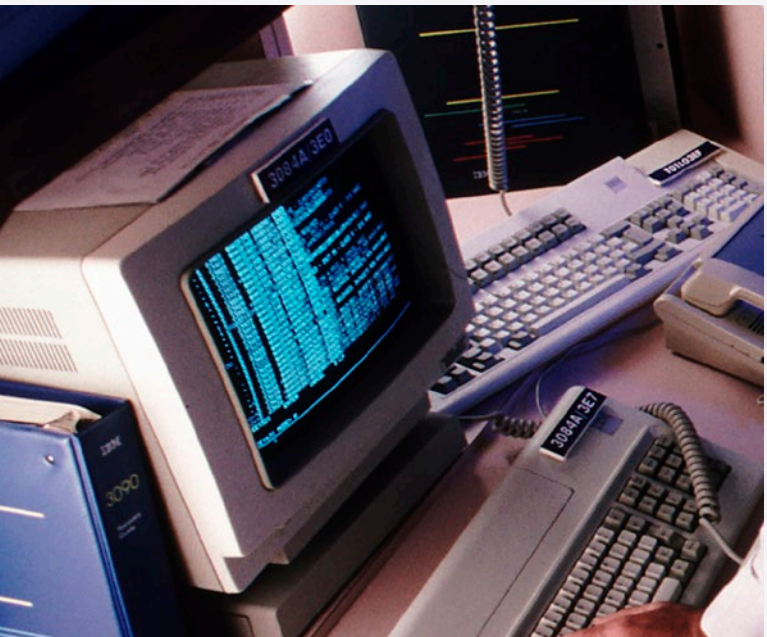
Om te begrijpen waarom dat zo radicaal was, moeten we terug naar de vroege decennia van de computer. Tot ver in de jaren vijftig draaide alles om hardware: buizen, transistors, schakelingen. Software was bijkomstig, een middel om de machine aan te sturen. Programmeurs waren pioniers die met kartonnen ponskaarten werkten, hun logica letterlijk gaatje voor gaatje vastlegden. Er werd nauwelijks gedocumenteerd, laat staan gestandaardiseerd. Het vak was meer improvisatie dan architectuur, meer kunst dan techniek.

De groei van systemen maakte dat onhoudbaar. In de Verenigde Staten probeerde de luchtmacht een landelijk commandosysteem te bouwen, NASA had software nodig voor de Apollo-raketten en in Europa kwamen grootschalige automatiseringsprojecten voor spoorwegen, telefonie en defensie op gang. Steeds vaker liep het mis: deadlines werden gemist, budgetten explodeerden en eenmaal opgeleverde systemen bleken nauwelijks te onderhouden. Software was onzichtbare infrastructuur: je merkte pas hoe kwetsbaar ze was als ze instortte.

Uit die opeenstapeling van mislukkingen ontstond een zoektocht naar structuur. In de decennia daarna volgden methodologische golven: het watervalmodel met zijn lineaire stappen van analyse tot implementatie, en later de agile- en scrumpraktijken met korte iteraties en feedbackloops. In de jaren negentig kwamen modellen die volwassenheid probeerden te meten, zoals CMMI. Elk van die benaderingen was een poging om grip te krijgen op iets wat zich telkens aan grip onttrok. Software engineering werd zo niet alleen een technische discipline, maar ook een culturele en organisatorische: de kunst om collectief te bouwen aan iets wat onzichtbaar bleef tot het draaide.

Die erfenis klinkt nog steeds door. Achter elk groot systeem van vandaag – medische apparatuur, financiële infrastructuren, luchtverkeersleiding – schuilt een traditie van normen, standaarden en controlemechanismen die zijn ontworpen om chaos te temmen. Software werd volwassen door regels, processen en gemeenschappelijke methoden.

En toch staan we nu opnieuw op een breukvlak. De zorg van toen was dat er te weinig orde was. De zorg van nu is dat orde volledig wordt overschaduwd door gemak.



AI-modellen genereren tegenwoordig code op basis van prompts in gewone taal. Je hoeft geen programmeur meer te zijn om applicaties te bouwen. De ontwikkelaar wordt prompter. Code wordt niet langer geschreven, maar georkestreerd – alsof ze vooraf al bestond, wachtend op de juiste vibe. Wat vroeger vakkennis vergde, vraagt nu vooral richting, toon en intentie. En wie die aanwijzingen formuleert, hoeft de onderliggende logica niet meer te begrijpen.

De term *vibe coding* werd begin 2025 geïntroduceerd door Andrej Karpathy en vond razendsnel ingang in de techwereld. De belofte was verleidelijk: beschrijf je intentie – de vibe van wat je wilt – en de AI bouwt het voor je. Geen syntaxis, geen architectuurdiagrammen, geen eindeloze *refactors*, alleen een gevoel, een richting, een prompt. De rest wordt gegenereerd.

Hoewel vibe coding zich primair richt op het genereren van broncode, reikt de impact ervan verder. Software engineering is immers veel meer dan alleen coderen: het omvat ook testen, documenteren, modelleren, onderhouden, samenwerken en beoordelen. Toch verandert juist dat eerste creatieve moment de rest van de keten. Waar eerder elke keuze expliciet en afweegbaar was, glijdt die nu onzichtbaar mee in de output van een prompt. De schijnbare eenvoud aan de voorkant maskeert de complexiteit aan de achterkant. En precies daar ontstaan de blinde vlekken.

Onder de belofte sluimeren daarom oude zorgen. Wie begrijpt de gegenereerde code nog? Wie garandeert de veiligheid, de schaalbaarheid, het onderhoud? Wat ge-

beurt er met vakmanschap als het vak zelf oplost in prompts? En belangrijker nog, hoe houden we grip op een infrastructuur die we zelf niet meer expliciet hoeven te ontwerpen?

Deze vragen zijn niet retorisch. Ze raken de kern van wat het betekent om technologie te gebruiken binnen complexe organisaties. Vibe coding maakt die spanning opnieuw zichtbaar: tussen snelheid en begrip, tussen innovatie en verantwoordelijkheid, tussen wat mogelijk is en wat wenselijk blijft. Dit rapport zoekt naar manieren om die spanning te begrijpen, te duiden en ermee om te gaan. Het kijkt naar de praktijk waarin vibe coding zich aandient, naar de risico's die het met zich meebrengt, naar de governance die nodig is om ermee te werken en naar de bredere gevolgen voor organisaties, werk en cultuur.

Wat in 1968 werd herkend als een crisis van improvisatie, verschijnt nu opnieuw – maar in een omgekeerde vorm. Destijds ging het om de noodzaak van structuur, vandaag om de noodzaak van begrip. Vibe coding is misschien de meest natuurlijke voortzetting van de digitale revolutie, maar juist daarom is het noodzakelijk opnieuw te vragen wat we willen bewaren.

In de volgende hoofdstukken werken we dat uit als verkenning van een landschap waarin software zich steeds meer laat schrijven door taal en waarin de mens opnieuw moet ontdekken wat het betekent om te blijven denken.

Hoofdstuk 2

Van ponskaart tot prompt

De lange evolutie naar vibe coding

Elke technologie begint met een belofte en met een vertaalslag. De belofte: dat machines ons werk verlichten, versnellen of verrijken. De vertaalslag: hoe brengen we menselijke bedoelingen over op een niet-menselijk systeem?

In de vroege dagen van de digitale revolutie werd deze vertaalslag letterlijk geponst in karton. Programmeurs gebruikten ponskaarten, waarin elk gaatje een instructie was, elke kaart een regel code. De logica was tastbaar, rigide en lineair. Wie een fout maakte, moest de hele kaart opnieuw ponsen. Wie een programma wilde begrijpen, moest het als een mechanisch artefact lezen: kaart voor kaart, regel voor regel.

Daarna kwamen hogere programmeertalen. Eerst assembler, toen Fortran, Pascal en C. Talen die de machine begreep, maar die al iets dichter bij menselijke denkpatronen lagen. Het was nog steeds een vak – moeilijk, abstract, gecodeerd – maar het denken werd iets natuurlijker. Niet langer in nullen en enen, maar in structuren, functies en logische blokken. De mens kreeg controle. En tegelijk groeide de afstand tot het onderliggende metaal.

De volgende sprong was interfacegericht: de grafische gebruikersomgeving. Denk aan Windows, de muis, drag-and-drop en visuele metaforen. Interactie werd beeld, handeling en beweging. Software werd gebruiksvriendelijker, maar ook complexer onder de motorkap. Een blauw scherm des doods was de finale confrontatie met deze complexiteit. Ondertussen verschoof de vertaalslag van code naar klik.

In de jaren 2010 en 2020 volgden *no-code* en *low-code* platforms: abstractie op abstractie. Platformen zoals Bubble, OutSystems en Mendix beloofden dat software gebouwd kon worden zonder code te schrijven. Een democratisering van ontwikkeling – maar nog steeds gebaseerd op sjablonen, drag-and-dropinterfaces en voorgeprogrammeerde componenten. De vertaalslag was vereenvoudigd, maar niet verdwenen.

En nu? We praten.

Met de komst van grote taalmodellen (LLM's) als GPT, Claude en Mistral is een nieuwe drempel genomen. Niet alleen het schrijven van code, maar ook het *begrijpen* ervan wordt overgenomen door AI. De gebruiker hoeft geen syntaxis meer te kennen, geen API-documentatie door te ploegen, geen bibliotheken te installeren. Een intentie volstaat. Een wens. Een vibe:

'Bouw een dashboard dat realtime temperatuurmetingen uit deze API haalt.'
De machine begrijpt en codeert.

Deze ontwikkeling is radicaal, maar niet onverwacht. Ze sluit aan bij visies van computervisionairs als Douglas Engelbart (de bedenker van de computermuis) en Alan Kay, die al in de jaren zeventig droomden van systemen die 'denken zoals mensen praten'. Niet omdat technologie dan menselijk wordt, maar omdat de mens dan vrijer wordt. Vrijer om zich te richten op ideeën, patronen en ontwerp – in plaats van syntaxis, foutmeldingen en standaardcode.

Maar hoe natuurlijker de interface, hoe onnatuurlijker het risico. Want als het systeem moeiteloos lijkt te begrijpen wat je bedoelt, wordt het verleidelijk te vergeten dat het ook fouten maakt. De prompt verhult de complexiteit. De illusie van begrip groeit. De zwarte doos wordt groter. De afstand tot de logica groeit.

Kortom:

Vibe coding is geen hype, maar het nieuwste stadium in een eeuwenoude interface-evolutie: van ponskaart naar prompt, van syntaxis naar semantiek, van instructie naar intentie.

Het is een vorm van coderen waarin het schrijven verdwijnt, maar het denken juist des te noodzakelijker wordt. En precies daarom verdient deze revolutie dezelfde kritische aandacht als de eerste.

Hoofdstuk 3

De geest van Garmisch-Partenkirchen

Wat de NATO Software Engineering Conference (1968) ons nog steeds leert

In september 1968 reisde een gezelschap van zestig wetenschappers, ingenieurs en militaire functionarissen af naar het Zuid-Duitse bergdorp Garmisch-Partenkirchen. De lucht was helder, de bergen waren stil, maar de reden voor hun komst was allesbehalve rustgevend. Er ging iets mis met software. Grootschalige automatiseringsprojecten, vaak met strategisch of militair belang, faalden op een manier die nieuw en verontrustend was. Ze liepen uit, liepen vast of leken op onverklaarbare wijze uit de hand te lopen. Niet omdat de hardware faalde, maar omdat de software zich gedroeg als een vreemd organisme: onvoorspelbaar, oncontroleerbaar en uiteindelijk onbegrijpelijk. De systemen groeiden, maar het inzicht kromp. Men begon te spreken van een softwarecrisis.

De conferentie, georganiseerd onder de vlag van de NAVO, was bedoeld als antwoord op die crisis. Niet met technische lapmiddelen of projectmanagementtips, maar met een fundamenteel voorstel: software moest een eigen ingenieursdiscipline worden. De organisatoren, onder wie informatiewetenschapper en computerpionier Brian Randell en de wiskundige, informaticus en Turing Award-winnaar Peter Naur, introduceerden daarom bewust een nieuwe term: *software engineering*. Dit in de hoop dat het vak zich zou gaan gedragen als een volwaardige ingenieurspraktijk, met dezelfde discipline, methodologie en kwaliteitsnormen als bruggenbouw, luchtvaart of elektrotechniek. Zonder kwaliteitsnormen en methodische standaarden, zo vreesden de aanwezigen, zou software geen oplossing zijn maar een nieuw soort wapen van chaos.

Daarbij ging het niet alleen om de code zelf, maar om het geheel van ontwerp, documentatie, onderhoud, samenwerking en toetsing: een discipline waarin systemen begrepen moesten kunnen worden – ook door anderen, ook na verloop van tijd.

Die keuze was provocatief. In het voorwoord van het conferentieverslag gaven de organisatoren toe dat het woord eerder een normatief streven dan een beschrijvende term was. 'We use the term not because it is an established reality,' schreven ze, 'but because it is a goal we must strive for.'

Het idee van software als ingenieursproduct was in 1968 nog revolutionair. Software werd tot dan toe vooral gezien als ambacht, als improvisatie, als het domein van briljante solisten die code schreven zoals een componist zijn partituur. Maar dat model hield geen stand in een wereld waarin systemen steeds complexer werden en waarin bedrijven, overheden en legers steeds afhankelijker raakten van onzichtbare, ondoorgrondelijke logica.

De problemen die tijdens de conferentie op tafel kwamen, klinken ook in 2025 nog verrassend bekend. Projecten faalden niet alleen door technische tekortkomingen, maar vooral door gebrek aan begrip. De logica van het systeem oversteeg het mentale model van de mens. Programmeurs konden elkaars werk niet lezen. Niemand wist waar fouten vandaan kwamen. En als iets werkte, wist niemand zeker waarom. Software had geen transparantie, geen toetsbaarheid, geen geheugen – alleen gedrag. En dat gedrag werd steeds moeilijker te controleren.

Wat de deelnemers aan de conferentie gemeen hadden, was een diep gevoel van ongemak. Niet alleen over de complexiteit van de systemen, maar ook over de cognitieve grenzen van de mens. De menselijke geest bleek niet ontworpen om duizenden regels code te overzien, laat staan hun onderlinge interacties, afhankelijkheden en neveneffecten. De Nederlandse informaticus Edsger Dijkstra, een van de invloedrijkste stemmen van zijn tijd, vatte het destijds kernachtig samen:

'The competent programmer is fully aware of the strictly limited size of his own skull.'

Die erkenning van onze beperking was het begin van een nieuwe ethiek. Een vak waarin je je verstand gebruikte om systemen te bouwen die je eigen verstand niet konden overbelasten. Een vak dat structuur eiste, standaarden, formaliteit. Een vak dat niet langer als kunstvorm, maar als verantwoordelijkheid moest worden beoefend.

Het is die ethiek die in 1968 werd geboren en die vandaag opnieuw onder druk staat. Want ook nu, meer dan vijftig jaar later, staan we op een technologisch breukvlak. Software wordt opnieuw complexer, maar ditmaal niet door menselijke hand. Tegenwoordig wordt code in seconden gegenereerd door grote taalmodellen, op basis van niets meer dan een paar zinnen in natuurlijke taal, getypt of simpelweg uitgesproken. Wat in 1968 nog een handmatig, gecontroleerd proces was, is in 2025 een onzichtbare samenwerking tussen mens en machine geworden. De ontwikkelaar schrijft geen code meer, hij beschrijft intenties. De machine vult aan. Vibe coding dus.

Toch raakt vibe coding slechts één schakel van het softwareproces: de expressieve vertaling van een idee naar werkende logica. De bredere principes van software engineering – zoals toetsbaarheid, schaalbaarheid, onderhoud, validatie en versiebeheer – blijven ook in deze nieuwe praktijk onverminderd relevant. Juist daarom is de vraag hoe deze AI-gegenereerde logica daarbinnen past, zo prangend.

Op het eerste gezicht lost vibe coding een aantal problemen op die in Garmisch-Partenkirchen al speelden. Het maakt softwareontwikkeling toegankelijker, sneller en meer iteratief. Goede en slechte code zal zich sneller over de wereld gaan verspreiden. Maar wie iets dieper kijkt, ziet hoe de oude zorgen in nieuwe vorm terugkeren. Want wie begrijpt de gegenereerde code nog? Wie garandeert de veiligheid, de robuustheid, de onderhoudbaarheid? En wie draagt de verantwoordelijkheid als systemen zich op onvoorziene manieren gaan gedragen?

In plaats van met handgemaakte spaghetticode, zoals in de jaren zestig, zitten we nu met machine-gegenereerde logica waarvan de oorsprong diffuus is en de impact moeilijk voorspelbaar. De zwarte doos is niet kleiner geworden, alleen gepolijster.

In het eindrapport van de conferentie staat een zin die, met de blik van nu, aan urgentie heeft gewonnen:

‘We should consider software to be a product to be engineered rather than a craft to be performed.’

Maar wat als de code straks niet meer *engineered* is, maar gegenereerd – en daarmee onttrokken aan het proces van expliciet ontwerpen, toetsen en begrijpen?

Dan is het misschien niet de software zelf die we moeten herzien, maar onze relatie tot software. En dan blijkt de geest van Garmisch-Partenkirchen geen historisch moment, maar een moreel kompas. Niet om terug te keren naar het verleden, maar om richting te houden in een toekomst die steeds sneller haar eigen logica schrijft.

Hoofdstuk 4

Vibe coding in de praktijk

Van belofte tot botsing

Het begon als een grap, een geinige tweet van Andrej Karpathy begin 2025: 'The hottest new programming language is English.' Een knipoog, zo leek het, naar de opkomst van grote taalmodellen die op commando hele functies konden uitschrijven. Maar binnen weken bleek het meer dan ironie. De term *vibe coding*, die Karpathy vrijwel gelijktijdig lanceerde, raakte een snaar in de ontwikkelgemeenschap. Prompten werd coderen. Intentie werd implementatie. En de *vibe* – het gevoel, de richting, het globale idee – werd voldoende om werkende software te genereren.

De revolutie voltrok zich razendsnel. Wat ooit gold als experimentele promptkunst voor hobbyisten, werd in korte tijd een volwaardige workflow. Nieuwe tools schoten als paddenstoelen uit de grond: *Mistral Code*, *Replit Ghostwriter*, *Cursor*, *v0.dev*, *CodeRabbit*. Elk van deze systemen belooft in essentie hetzelfde: dat de gebruiker de machine slechts hoeft te vertellen *wat* hij wil, niet *hoe*. Een enkele zin volstaat. Een ruwe schets, een halfbakken idee, een globale instructie – en de machine vult aan, vervolledigt, vertaalt. Terwijl Sineks Golden Circle organisaties opriep om te beginnen bij het 'waarom', lijkt vibe coding te functioneren als een *inverse cirkel*: de gebruiker begint bij het 'wat', de machine genereert het 'hoe' en het 'waarom' raakt diffuus. Daarmee komt de autonomie van de maker onder druk te staan: als de machine het werk doet, waar ligt dan nog het kompas?

Deze stijl van werken is radicaal anders dan de klassieke softwareaanpak. Geen lange requirements-documenten meer, geen wireframes, geen specificaties vooraf. In plaats daarvan ontstaat code al doende, in dialoog tussen gebruiker en model. De workflow is iteratief, associatief, improviserend. Je probeert iets, test het, herformuleert je prompt en laat de machine opnieuw beginnen. De menselijke rol verschuift van maker naar curator. Niet bouwen, maar begeleiden. Niet schrijven, maar schaven.

Voor kleine projecten werkt dit verrassend goed. Hobbyisten bouwen binnen een uur een receptenapp, een scraper of een chatbot. Start-ups gebruiken vibe coding om sneller dan ooit MVP's (Minimum Viable Products) te lanceren. Volgens een rapport[1] van *TechCrunch* werd in het winterprogramma van Y Combinator – een bekende Amerikaanse start-upversneller – bij meer dan een kwart van de jonge technologiebedrijven 90 procent of meer van de softwarecode automatisch gegenereerd door AI in plaats van handmatig geschreven door programmeurs. Vroege surveys ondersteunen dit beeld: platforms als Replit melden dat organisaties die AI-gestuurd ontwikkelen, tot bijna zes keer sneller applicaties kunnen opleveren, terwijl Y Combinator-partners zelfs spreken van een honderdvoudige versnelling bij sommige start-ups. Zulke voorbeelden laten zien hoe sommige organisaties zelfs complete MVP's volledig uit AI laten rollen, met snelheden die eerder ondenkbaar waren, maar ook met een duidelijk hogere kans op bugs en kwetsbaarheden die pas later aan het licht komen.

De journalist Kevin Roose beschreef in *The New York Times*[2] hoe hij in één avond een app bouwde die suggesties deed op basis van de inhoud van zijn koelkast. 'Ik typte simpelweg in: "Maak een webapp die mijn boodschappenlijst analyseert en suggesties doet voor lunchgerechten." Binnen enkele minuten had ik een functionerend prototype.' Pas later ontdekte hij dat het model verzonnen ingrediënten gebruikte en valse e-commerce-recensies simuleerde. De interface werkte perfect, de waarheid niet.

Ook meer geavanceerde toepassingen duiken op. In de *DEV Community* beschreef een ontwikkelaar hoe hij een webapp bouwde genaamd *ResearchAI Assistant*,[3] die wetenschappelijke papers omzet in podcasts, visuele slides en TikTok-samenvattingen. De hele applicatie werd gebouwd met vibe coding tools, zonder dat de maker zelf diepgaande ervaring had in backend development.

[1] TechCrunch. (2025, 30 juni). *A comprehensive list of 2025 tech layoffs*. https://techcrunch.com

[2] Roose, K. (2025, 27 februari). Not a Coder? With A.I., Just Having an Idea Can Be Enough. *The New York Times*. https://www.nytimes.com/2025/02/27/technology/personaltech/vibecoding-ai-software-programming.html

[3] Deb, A. (2024, 9 november). *Research AI Assistant Application – RAG*. DEV Community. https://dev.to/ayushdeveloper/research-ai-assistant-application-rag-3d0m

Het enthousiasme is begrijpelijk. Vibe coding democratiseert een domein dat lange tijd gesloten was voor leken. Het verlaagt de drempel tot creatie, versnelt prototyping en verkort de afstand tussen idee en implementatie. Voor de creatieve industrie, het onderwijs, de start-upwereld en zelfs interne tools in grotere organisaties biedt het ongekende snelheid en flexibiliteit. Non-tech founders grijpen de kans om hun ideeën rechtstreeks te prototypen en online circuleren inmiddels challenges als '12 projecten in 12 maanden' of zelfs '24 projecten in 24 uur', waarmee vibe coding ook een cultureel fenomeen wordt dat de belofte van democratisering tastbaar maakt.

Een nog veelzeggender teken van die culturele verschuiving is de opkomst van Chad IDE, door de makers gepresenteerd als 'de brainrot-IDE'. Het uitgangspunt is veelbetekenend: terwijl de AI code genereert, kunnen developers de wachttijd in dezelfde omgeving vullen met minigames, korte videofeeds en andere vormen van entertainment. Wat begon als een workflowoptimalisatie, verandert zo in iets cultureels: een IDE die niet alleen programmeren ondersteunt, maar ook afleiding reguleert. Chad IDE klinkt misschien als satire, maar juist daardoor legt het een dieper inzicht bloot over vibe coding. Wanneer softwaregeneratie snel genoeg gaat om magisch te voelen, maar traag genoeg blijft om micromomenten van verveling te laten ontstaan, wordt zelfs wachten een ontwerpvraagstuk. En zodra die magie slijt, maken die micro-gaps plaats voor een fundamentelere vraag: wat bouwen gebruikers eigenlijk en hoe goed begrijpen ze het nog?

Maar die belofte botst steeds vaker met de weerbarstige praktijk. Want waar de tool intuïtief lijkt, is de onderliggende code dat vaak allerminst. Veel vibe-generated code werkt oppervlakkig gezien goed, maar bevat kwetsbaarheden, inconsistenties of verborgen afhankelijkheden. Debuggen is moeilijk – je kunt immers niet debuggen wat je niet begrijpt. Ook schaalbaarheid is een probleem. Wat als de app plots miljoenen gebruikers krijgt? Wie herschrijft de code die door niemand is geschreven?

En dan is er de kwestie van eigenaarschap. Wie is verantwoordelijk voor AI-gegenereerde code? De promptschrijver? De engineer die ernaar keek? De organisatie die de code gebruikte? Of de maker van het taalmodel dat de code genereerde? In juridische zin is het antwoord nog onzeker. In praktische zin evenzeer. Want wat vandaag werkt, is morgen onbegrijpelijk – zeker als niemand weet waar de logica vandaan kwam.

In de praktijk leidt dit tot een fundamentele verschuiving in het denken over softwarekwaliteit. Niet langer is het product de resultante van menselijke precisie, maar van stochastische waarschijnlijkheid. Het model doet een gok – een weloverwogen gok, op basis van miljarden parameters, maar toch een gok. En die gok wordt vaak niet meer geverifieerd. De gebruiker ziet dat het werkt en gaat door. De illusie van correctheid wint het van de toets van begrip.

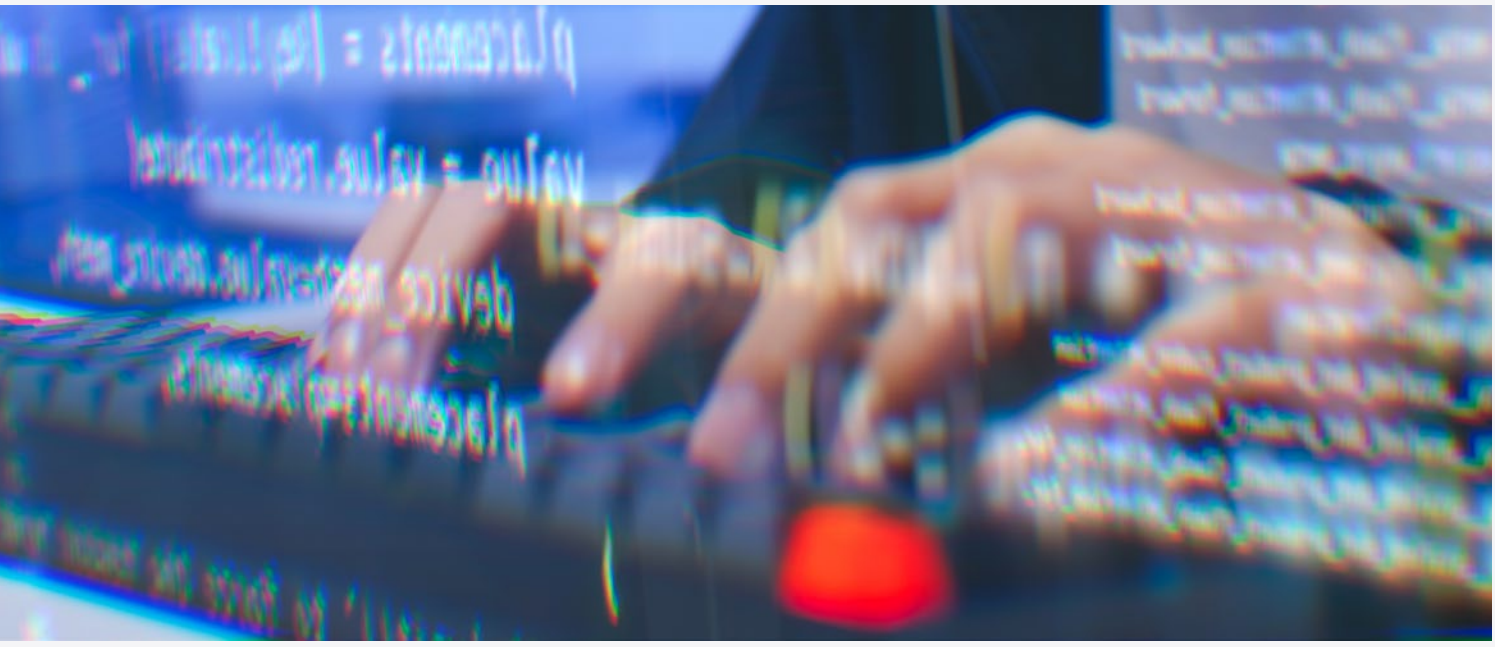
Voor CIO's en CTO's die overwegen vibe coding te integreren in hun organisatie, is de aantrekkingskracht groot, maar het risico eveneens. Want hoe beoordeel je software als

de logica ervan buiten je cognitieve bereik ligt? Hoe onderhoud je een systeem dat door niemand écht is begrepen? Hoe bouw je voort op een *code base* die niet is geschreven, maar ontstaan?

De botsing tussen belofte en realiteit is niet fataal, maar vraagt om bewust ontwerp. Niet alleen van tools, maar van organisatiestructuren, opleidingsmodellen en governancepraktijken. Vibe coding is geen hype, maar een nieuwe taal, en elke nieuwe taal vraagt om grammatica, om normen, om collectieve interpretatie.

Wie deze taal serieus neemt, ziet dat ze niet alleen over code gaat, maar ook over samenwerking. Vibe coding dwingt tot een herziening van de verhouding tussen mens en machine: wie bepaalt de richting, wie levert de betekenis, wie leert van wie? In die verschuiving schuilt een onderscheid dat zelden wordt uitgesproken, maar bepalend is voor wat vibe coding werkelijk betekent.

Sommige toepassingen van AI nemen het werk letterlijk over – een vorm van **automation**, waarin de mens nog hooguit toekijkt. Andere versterken het werk van de mens – een vorm van **augmentation**, waarin AI een partner wordt in denken, ontwerpen en testen. Dit onderscheid werd scherp geformuleerd in het *Anthropic Economic Index*-rapport[4] over de impact van AI op softwareontwikkeling: *automation* verwijst naar 'AI systems that directly perform the task end-to-end', terwijl *augmentation* duidt op 'AI systems that collaborate with a user to complete a task'. Volgens dezelfde analyse is in Claude Code ongeveer **79 procent** van de coderingsinteracties geclassificeerd als automation, tegenover **49 procent** bij Claude.ai – een verschil dat precies laat zien hoe de aard van samenwerking de diepte van begrip bepaalt.



In de eerste variant verschuift verantwoordelijkheid naar het systeem, in de tweede blijft zij gedeeld. Automation creëert gemak, augmentation onderhoudt begrip.

Tussen die twee uitersten ontvouwt zich echter nog een derde toestand, die in recente onderzoeksliteratuur wordt aangeduid als **scaffolding**. Daarmee wordt bedoeld dat AI een tijdelijk kader neerzet waarin de mens kan handelen, leren of ontwerpen. De machine bouwt een steiger van suggesties, structuur en richting; de mens beklimt die om eigen inzicht te vormen. Zolang de steiger zichtbaar blijft, behouden we overzicht. Pas wanneer de steiger onzichtbaar wordt – wanneer we vergeten dat die slechts tijdelijk was – verandert hulp in overname.

In dat schemergebied tussen begeleiding en vervanging zal de toekomst van softwareontwikkeling zich waarschijnlijk afspelen: hoe houden we de mens denkend aanwezig in een infrastructuur die steeds meer zelf uitvoert?

Recente metingen onder ervaren softwareontwikkelaars laten zien hoe precair dat evenwicht in de praktijk is. In een gecontroleerd experiment[5] met de nieuwste generatie generatieve modellen bleken programmeurs hun taken gemiddeld 19 procent langzamer te voltooien mét AI dan zonder, terwijl ze hun eigen productiviteit juist hoger inschatten. Wat zichtbaar wordt, is geen technische tekortkoming, maar een cognitieve: de snelheid van suggestie maskeert de traagheid van begrip. De machine reduceert frictie, maar niet inspanning; ze verschuift de arbeid van uitvoering naar verificatie. Wat resteert is het gevoel van versnelling zonder de ervaring van vooruitgang.

In de volgende hoofdstukken zullen we dieper ingaan op de risico's, de organisatorische implicaties en het strategisch handelingsperspectief dat nodig is om vibe coding verantwoord te integreren in volwassen ontwikkelomgevingen. De vraag is niet of deze stijl van softwareontwikkeling zal blijven, maar wie het gesprek erover blijft voeren.

[4] Anthropic. (2025, 28 april). *Anthropic Economic Index: AI's impact on software development.* Anthropic Research. https://www.anthropic.com/research/impact-software-development

[5] Becker, J., Rush, N., Barnes, B., & Rein, D. (2025, juli). *Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity.* METR Working Paper. https://arxiv.org/pdf/2507.09089

Hoofdstuk 5

Drie verhalen over agentische versnelling

De voorgaande hoofdstukken beschreven hoe vibe coding zich ontwikkelt van experimentele promptkunst tot volwaardige ontwikkelpraktijk. Toch blijft de vraag wat dit in de dagelijkse realiteit betekent. Hoe ziet het eruit wanneer teams daadwerkelijk leren bouwen met AI als partner in plaats van als tool?

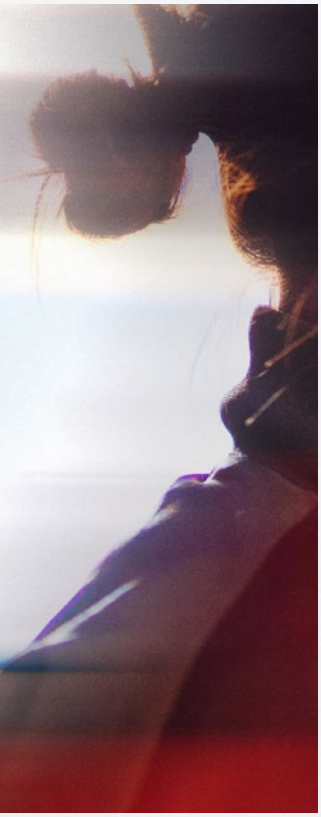
De volgende drie cases, afkomstig uit de Verenigde Staten, laten zien hoe dat samenspel zich in de praktijk ontvouwt. Ze tonen niet alleen een technologische versnelling, maar ook een verschuiving in energie, eigenaarschap en werkritme.

De retailer die zijn ritme vond

Bij een Amerikaanse retailer met een jaaromzet van 12 miljard dollar verliep softwareontwikkeling altijd volgens vaste patronen. Updates, integraties en interne tools werden zorgvuldig gepland, betrouwbaar maar traag. Het engineeringteam werkte gestructureerd, maar zelfs kleine verbeteringen vergden dagenlang handmatig testen en controleren. Toen het team besloot een proef te doen met agentische workflows, waren de verwachtingen bescheiden.

De eerste weken was het een kwestie van aftasten. Ontwikkelaars moesten leren omgaan met AI-agenten die testcases konden genereren, fouten opspoorden en stukken code schreven. Het voelde aanvankelijk onwennig, alsof men een onvermoeibare junior ontwikkelaar moest begeleiden. Maar gaandeweg ontstond een nieuwe dynamiek: minder tijdverlies, meer concentratie. Taken die vroeger één of twee dagen vergden, werden in enkele uren afgerond.

Wat het team vooral verraste, was niet de snelheid maar de energie. Engineers gingen elkaar helpen, nieuwe ideeën dienden zich aan en de sfeer veranderde. Wat begon als een voorzichtig experiment, groeide uit tot iets blijvends. Inmiddels past het bedrijf de methode toe in meerdere productlijnen – niet omdat het moet, maar omdat niemand nog terug wil naar hoe het was.



De fabrikant die een crisis omboog tot momentum

Een Amerikaanse fabrikant had acht maanden gewerkt aan een complexe mobiele app, gebouwd met een populair low-code platform. Ondanks de inzet liep het project vast. De prestaties vielen tegen, integraties mislukten en het vertrouwen slonk. Met nog vijf weken budget over stond de projectleider voor een lastige keuze: stoppen of iets radicaal anders proberen.

Ze besloten opnieuw te beginnen, dit keer met een traditioneel raamwerk, ondersteund door *agentic development*. Al snel veranderde de sfeer. AI-agenten namen repetitieve taken over, bouwden modules en testten integraties, terwijl de ontwikkelaars zich konden richten op structuur en samenhang.

Niet alles ging vanzelf. Er waren momenten van verwarring, van te veel vertrouwen op de AI, van handmatig herstelwerk. Maar het tempo en de helderheid waarmee het team leerde, waren opvallend. Na vijf weken was de app af. Een project dat verloren leek, werd een kantelpunt. Tijdens de evaluatie besefte men dat de grootste winst niet het product was, maar de manier waarop het team had leren werken: sneller, rustiger, met meer overzicht.

Het onderwijsbedrijf dat zijn eigen tijdlijn herschreef

Een onderwijsinstelling met een verouderd platform stond voor een ander probleem: niet bouwen, maar vernieuwen. De infrastructuur was log, onderhoud was kostbaar en innovatie kwam nauwelijks van de grond. De directie schatte dat een volledige herbouw zeker een jaar zou duren – te lang voor een organisatie die snel moest meebewegen.

Het team besloot agentic development te gebruiken als proef. Al in de eerste week zagen ze verschil. AI-agenten analyseerden de bestaande code, brachten afhankelijkheden in kaart en stelden routes voor modernisering voor. Ontwikkelaars konden zich

richten op ontwerp, gebruikerservaring en architectuur. Het werk voelde minder als herschrijven van code, maar meer als samenwerken met een extra team dat nooit moe werd.

Na 6 weken was 70 procent van de applicatie overgezet naar een moderne stack. Het resultaat was niet alleen technisch, maar ook cultureel. De teams zagen wat er mogelijk was wanneer menselijke expertise en AI-precisie elkaar aanvullen. De ervaring leerde dat modernisering niet per se tijdrovend hoeft te zijn – als je de tijd op een andere manier leert gebruiken.

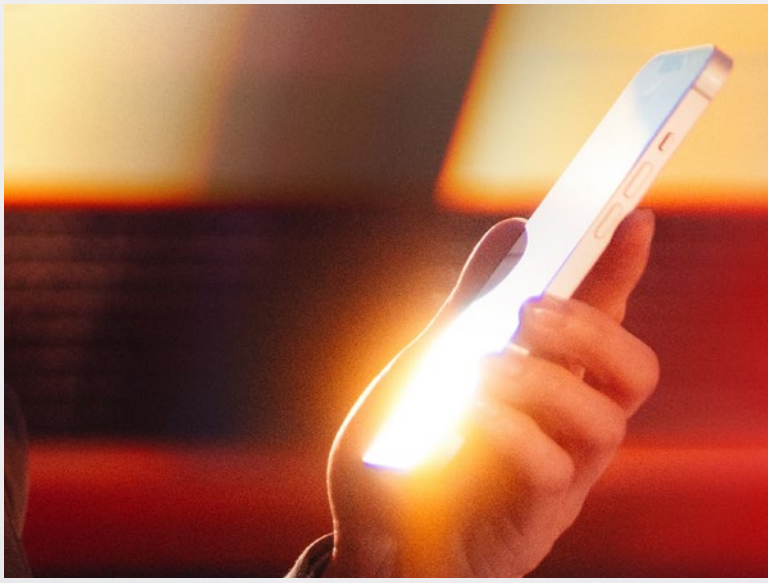
Een nieuw bouwritme

Elk van deze verhalen begon als een experiment en eindigde als een inzicht. Wat ze verbindt, is niet alleen de versnelling, maar het herwonnen gevoel van richting. Agentic development verkort tijd, maar vergroot ook de ruimte voor denken. Ontwikkelaars verschuiven van uitvoerders naar ontwerpers van samenwerking. De routine verdwijnt, de nieuwsgierigheid keert terug.

In deze bedrijven voelt vooruitgang niet langer als een strijd tegen de klok, maar als werken mét de tijd. Dat is de stille kracht van agentische versnelling: wanneer mens en AI samen bouwen, wordt het werk niet alleen sneller, maar ook beter, helderder en betekenisvoller.

Deze drie praktijkverhalen laten zien dat de belofte van vibe coding niet alleen in snelheid ligt, maar ook in het ontstaan van een nieuw werkritme waarin mens en model elkaar versterken. Toch roept dat succes een andere vraag op: wat verdwijnt er terwijl we versnellen?

In het volgende hoofdstuk onderzoeken we de keerzijde van dit gemak: de risico's, afhankelijkheden en blinde vlekken die zich schuilhouden achter de vloeiende samenwerking tussen mens en machine.



Hoofdstuk 6
De keerzijde van gemak

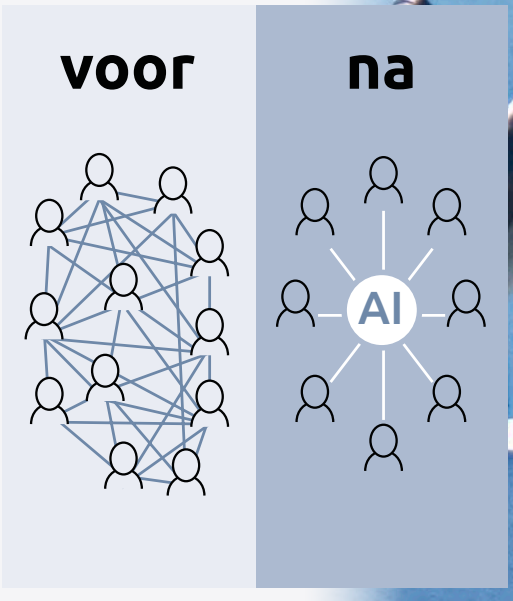
Risico's, kwetsbaarheden en het verdwijnen van verantwoordelijkheid

De programmeur wordt promptschrijver, de ontwikkelaar wordt regisseur van een conversatie. Waar voorheen syntaxis, compilerfouten en *dependency conflicts* dagelijks werk waren, lijkt nu vooral intuïtie voldoende. Je beschrijft wat je wilt, de machine doet de rest. De wereld van code wordt een wereld van taal.

Maar onder de oppervlakte van dit gemak liggen spanningen. Want terwijl de AI moeiteloos code genereert, neemt het aantal mensen dat die code begrijpt juist af. En met dat begrip verdwijnt ook iets fundamenteels: verantwoordelijkheid. Wat vibe coding oplost in snelheid, offert het op aan diepgang. Wat het democratiseert, verwaarloost het in controle. En wat het mogelijk maakt, laat zich niet altijd meer corrigeren.

Wij merken het zelf dagelijks: we communiceren minder met onze collega's dan voorheen. Niet omdat we ze ontwijken, maar omdat we ze eenvoudigweg niet meer nodig hebben voor de meeste vragen. Waar je vroeger langs hun bureau liep of een overleg inplande, open je nu een venster met AI en krijg je binnen een paar seconden een antwoord. Het voelt efficiënt, maar ongemerkt verschraalt het weefsel van samenwerking.

Een eenvoudige tekening maakt dit zichtbaar. Aan de linkerkant staat het oude patroon: een wirwar van lijnen tussen mensen, rommelig maar rijk, vol toevallige verbanden, fricties en gezamenlijke afwegingen. Aan de rechterkant verschijnt de nieuwe constellatie: iedereen rechtstreeks verbonden met AI, keurig overzichtelijk, maar zonder horizontale lijnen tussen elkaar. De ruis verdwijnt, maar met de ruis verdwijnt ook de dialoog.



Juist die horizontale verbindingen zijn de plekken waar kennis wordt getoetst, waar misverstanden aan het licht komen, waar vertrouwen groeit en verantwoordelijkheid wordt gedeeld. Door gemak te kiezen, lopen we het risico dat het sociale weefsel van organisaties oplost tot een verzameling individuen die ieder hun eigen gesprek met een machine voeren.

Onzichtbare risico's en technische blinde vlekken

Een van de grootste risico's is *onzichtbare complexiteit*. AI-code is zelden modulair, elegant of gedocumenteerd. De code is vaak voldoende om te werken, maar niet voldoende om later te onderhouden of aan te passen. De code 'loopt', maar niemand weet waarom precies. Dat maakt het lastig om fouten te corrigeren, om schaalproblemen op te lossen of om nieuwe ontwikkelaars in te werken. De code base wordt een soort syntactisch sediment: laag op laag, opgebouwd uit goedbedoelde prompts, maar zonder overzicht, structuur of consistentie.

Daar komt bij dat veel gegenereerde code onbewust fouten bevat. Niet omdat het model onkundig is, maar omdat het werkt met waarschijnlijkheden, niet met garanties. Taalmodellen zijn getraind om het meest waarschijnlijke antwoord te geven op een vraag en niet per se het meest veilige, schaalbare of robuuste antwoord. Ze imiteren patronen uit hun trainingsdata, inclusief achterhaalde of foutieve codevoorbeelden. Wat eruitziet als werkbare logica, kan in werkelijkheid gebaseerd zijn op verouderde databanken, onveilige praktijken of simplistische aannames. Een overzichtsstudie uit 2025[6] van vibe-codingpraktijken laat zien dat snelheid en kwaliteit niet altijd hand in hand gaan. In meerdere onafhankelijke observaties wordt gesproken van een duidelijke speed-versus-quality trade-off: projecten worden aanzienlijk sneller opgeleverd, maar de gegenereerde code bevat vaker inconsistenties, onderhoudsproblemen of verborgen fouten. De eerste versie van AI-gegenereerde code is meestal 'fast but flawed': functioneel werkbaar, maar zelden productierijp zonder extra kwaliteitscontrole. Deze vroege bevindingen wijzen erop dat ontwikkelteams die met vibe coding werken, meer tijd besteden aan refactoring en kwaliteitsherstel dan bij conventionele ontwikkeling. De belofte van versnelling blijft reëel, maar vraagt om evenredige aandacht voor toetsing en onderhoud.

Vibe coding raakt echter maar een deel van wat software engineering is. Het laat zien hoe code sneller kan ontstaan, maar software engineering is meer dan dat. Het is ook het proces van het achterhalen van behoef-

[6] Fawzy, A., Tahir, A., & Blincoe, K. (2025). *Vibe Coding in Practice: Motivations, Challenges, and a Future Outlook – A Grey Literature Review. arXiv.* https://arxiv.org/abs/2510.00328

ten, het ontwerpen van architecturen, het testen en onderhouden van systemen en het bewaken van veiligheid en schaalbaarheid. In die bredere discipline is vibe coding hooguit een versneller in de implementatiefase, niet het antwoord op de volledige opgave. Dat contrast is wezenlijk, want juist in de ruimte tussen het snelle bouwen en het langdurig onderhouden liggen de meeste uitdagingen van ons vak.

De gevolgen van deze cognitieve en technische blinde vlekken zijn al zichtbaar. In maart 2025 werd het vibe-codingplatform Lovable gewaarschuwd dat 170 van hun 1645 gegenereerde apps gebruikersdata lekten: namen, e-mails, financiële informatie en API-sleutels. De affaire voedde een bredere golf van kritiek: verschillende blogs bestempelden vibe coding halverwege 2025 als 'the worst idea of 2025',[7] waarbij ervaren developers waarschuwden dat juist juniors worden verleid door de snelheid, zonder de kennis om de risico's te overzien. Twee maanden later waren de kwetsbaarheden nog steeds niet opgelost – een open uitnodiging voor hackers om gebruikers financieel te ruïneren. Een andere zelfbenoemde vibe coder bouwde een SaaS-product met 'zero hand-written code' – tot bleek dat betalende gebruikers de abonnementen konden omzeilen, API-limieten werden overschreden en de database corrupt raakte. Zijn verklaring? 'Ik ben niet technisch, dus het kost me wat langer om het op te lossen.' En op Reddit waarschuwen beveiligingsexperts voor een nieuw soort cyberaanval die speciaal ontstaat door vibe coding. AI-modellen die code genereren, verzinnen soms bibliotheken of stukjes software die helemaal niet bestaan, maar die wel lijken op

[7] Parker, E. (2025, 26 augustus). *Vibe Coding Is the Worst Idea of 2025 – Here's Why It Fails.* Land of Geek. https://www.landofgeek.com/posts/vibe-coding-worst-idea-2025

echte. Kwaadwillenden spelen daarop in door die verzonnen namen daadwerkelijk te registreren maar dan met schadelijke code erin. Zo'n aanval heet *slopsquatting*. Wanneer een onervaren vibe coder zo'n fout advies klakkeloos overneemt en de gesuggereerde code installeert, haalt hij onbedoeld een achterdeur binnen voor hackers. Een klein foutje in een prompt kan zo leiden tot een aanval die het hele systeem besmet.

Vervagend eigenaarschap en juridische leegte

Dit wordt nog problematischer in sectoren waar compliance, veiligheid of aansprakelijkheid centraal staat. In een banksysteem, een zorgapplicatie of een defensiesysteem volstaat het niet dat de code *werkt*, het moet helder zijn *waarom* het werkt, *wanneer* het faalt en *wie* verantwoordelijk is als het fout gaat. In veel vibe-codingprojecten ontbreekt die herleidbaarheid. De logica is nergens vastgelegd, de oorsprong is niet reproduceerbaar en het gedrag van de software wordt pas zichtbaar bij gebruik.

Daarmee komen we bij een derde, fundamenteler punt: het vervagen van eigenaarschap. In de klassieke softwarepraktijk is het duidelijk wie verantwoordelijk is voor een module, een release of een patch. Maar bij vibe coding wordt de grens tussen gebruiker en systeem vager. De ontwikkelaar stelt een prompt op, maar de uitvoering ervan – de concrete implementatie in code – wordt gedaan door een model dat nergens formeel verantwoordelijk voor is. Wie 'schrijft' dan nog de software? En wie draagt de juridische last als iets misgaat?

Deze vraag is nog nauwelijks beantwoord, ook niet juridisch. Is AI-code auteursrechtelijk beschermd? Is de promptschrijver aansprakelijk? Of de organisatie? Of de maker van het taalmodel? In afwezigheid van duidelijke kaders ontstaat er een vacuüm waarin systemen operationeel zijn zonder dat iemand verantwoordelijk is voor hun gedrag. Een softwarepraktijk zonder aansprakelijkheid – iets waar de deelnemers van de NAVO-conferentie in 1968 voor waarschuwden, lang voordat AI bestond.

Een (fictieve) brief aan de moderne ontwikkelaar

Beste ontwikkelaar,

U werkt in een tijd waarin machines code kunnen schrijven op basis van natuurlijke taal, waarin u 'vibe codet' – intuïtief, creatief en zonder dat u iedere regel zelf hoeft te schrijven. U beschrijft uw intentie en de machine vult aan. Een technologische droom, zonder twijfel.

Maar toch wil ik u een vraag stellen. Wie draagt er verantwoordelijkheid voor deze gegenereerde code? Wie begrijpt zijn structuur, zijn fouten, zijn gedrag onder druk? In 1968 stonden wij aan de wieg van wat we toen trots 'software engineering' noemden. Niet omdat het al een rijp vakgebied was, maar omdat het die status dringend nodig had.

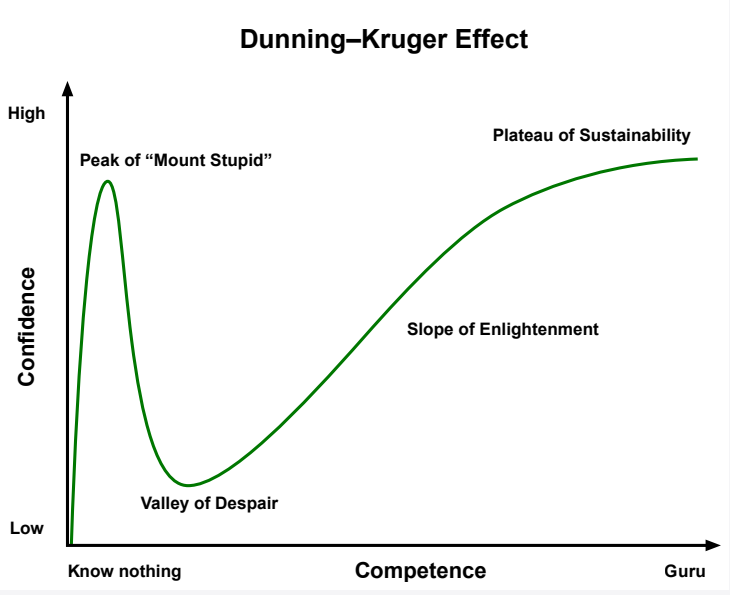
Onze zorg was simpel: zonder duidelijke methoden, standaarden en verantwoordelijkheid zou slechte software zich verspreiden als onkruid door systemen, organisaties en samenlevingen. Vandaag, nu software zichzelf lijkt te schrijven, is die zorg relevanter dan ooit.

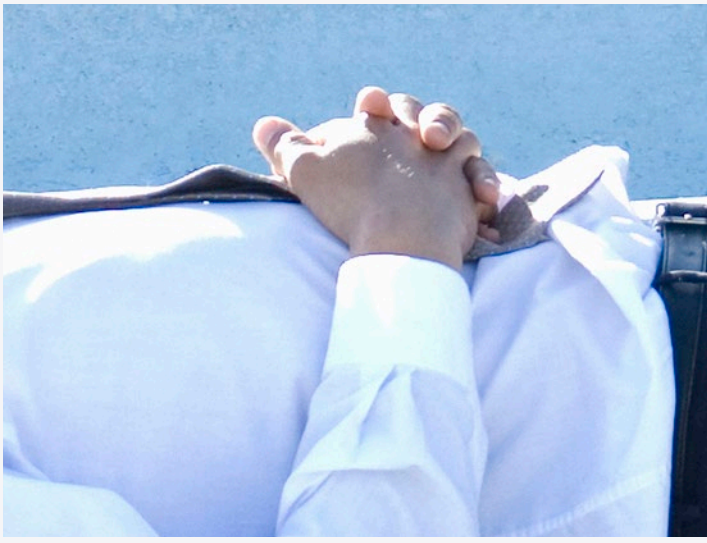
Met vriendelijke groet,

Brian Randell, co-editor NATO Software Engineering Conference, 1968

De illusie van beheersing

Daarbovenop speelt een psychologisch risico dat zelden wordt benoemd: het Dunning-Kruger-effect. Deze cognitieve bias, voor het eerst empirisch beschreven door psychologen David Dunning en Justin Kruger in 1999, stelt dat mensen met geringe competentie in een bepaald domein hun eigen kunnen systematisch overschatten – juist omdat ze het inzicht missen om hun tekortkomingen te herkennen. In de context van vibe coding is dit effect buitengewoon relevant. De interface is vriendelijk, de output indrukwekkend en het resultaat direct bruikbaar. Maar schijn bedriegt. Juist omdat de tooling zo laagdrempelig is, groeit de illusie van beheersing sneller dan de feitelijke expertise. Dit effect is niet alleen psychologisch, maar ook economisch zichtbaar: sommige bedrijven rapporteerden *runaway costs*, projecten die in eerste instantie goedkoop en snel leken maar uiteindelijk duurder uitvielen door onverwachte herbouw, verborgen bugs en oplopende onderhoudskosten. Je hoeft niet te weten hoe een algoritme werkt om het te gebruiken – en precies daarin schuilt het risico.





Vibe coding laat gebruikers opereren op een semantisch niveau dat losstaat van de onderliggende technische realiteit. Je hoeft niet te begrijpen wat een compiler doet, hoe geheugenbeheer werkt of hoe API's elkaar aanroepen. Maar zodra de gegenereerde code faalt – subtiel of spectaculair – ontbreekt de kennis om oorzaken te achterhalen of gevolgen te overzien. De combinatie van een hoge mate van gebruiksgemak en een lage mate van zelfinzicht creëert een gevaarlijke asymmetrie: gebruikers worden overmoedig, terwijl hun beslissingen onderbouwd worden met schijnkennis.

De gevaren zijn niet louter theoretisch. In meerdere organisaties is al gebleken dat junior medewerkers die zich bediend voelen door AI-tools, zich sneller opwerpen als autonome ontwikkelaars – zonder de ervaring om patronen van slechte code of structurele risico's te herkennen. De tooling fluistert bevesti-

ging, geen correctie. En wie door AI bekrachtigd wordt in zijn intuïtie, is des te minder geneigd om die intuïtie te bevragen. Het gevolg is niet alleen een verhoogd risico op fouten, maar ook een afname van lerend vermogen: feedbackloops vervagen, omdat het zelfvertrouwen harder groeit dan de kennis.

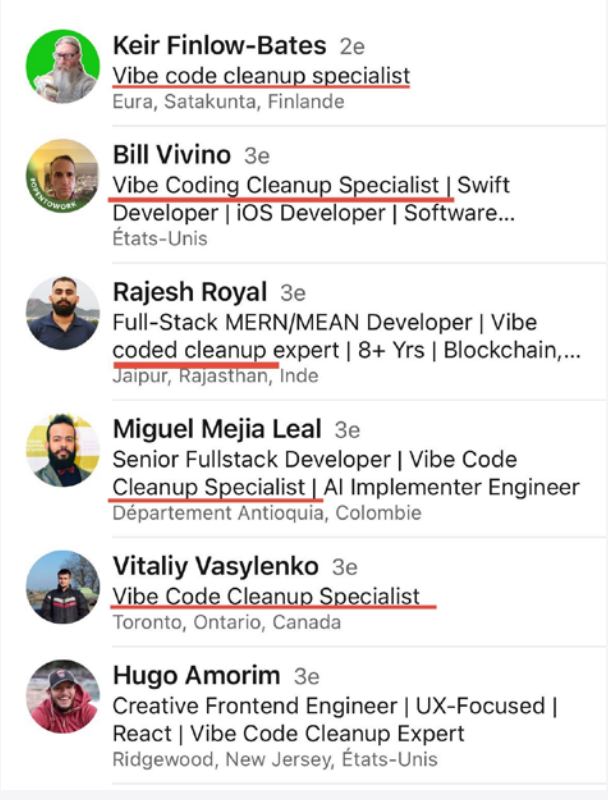
Deze psychologische valkuil verdient daarom expliciete aandacht in elk governancemodel rondom vibe coding. Niet alleen technische toetsing, maar ook educatie, bewustwording en teamdynamiek zijn essentieel. De grootste kwetsbaarheid zit immers niet in het model, maar in de mens die denkt dat hij het beheerst.

Op de achtergrond speelt ook nog een langetermijnvraag die technischer is, maar niet minder belangrijk: wat doet vibe coding met onze collectieve kennisstructuren? Als nieuwe generaties ontwikkelaars opgroeien met tools die alles voor hen schrijven, verliezen ze dan het vermogen om diep te begrijpen? Wordt coderen een ritueel, geen vaardigheid meer – een liturgie van prompts in plaats van logica? En als dat zo is: wie kan dan nog bouwen aan de systemen van de toekomst?

De zorgen zijn niet hypothetisch. Ervaren ontwikkelaars vergelijken vibe coding regelmatig met het bouwen van een huis uit Lego-blokjes: het oogt indrukwekkend, maar niemand weet zeker of het blijft staan. De vergelijking is simplistisch, maar raakt een kern van waarheid. Vibe coding creëert een schijn van beheersing – systemen zien er af en toe perfect uit – terwijl de structurele logica eronder vaak ondoorzichtig blijft.

Een nieuwe ethiek voor gegenereerde software

Toch is het niet nodig om te vervallen in pessimisme. Vibe coding *kan* veilig, transparant en productief worden ingezet, mits organisaties bereid zijn om nieuwe vormen van governance, kwaliteitscontrole en verantwoordelijkheid te ontwikkelen. Het vraagt om nieuwe rollen (zoals promptarchitect of AI-verificateur), nieuwe processen (waaronder code-review van gegenereerde blokken) en bovenal: een nieuw ethisch kader. We moeten op zoek naar een manier om de geest van Garmisch-Partenkirchen opnieuw te activeren in een wereld waarin software niet langer alleen *geschreven*, maar ook *gegenereerd* wordt.



Hoofdstuk 7

Governance en verankering

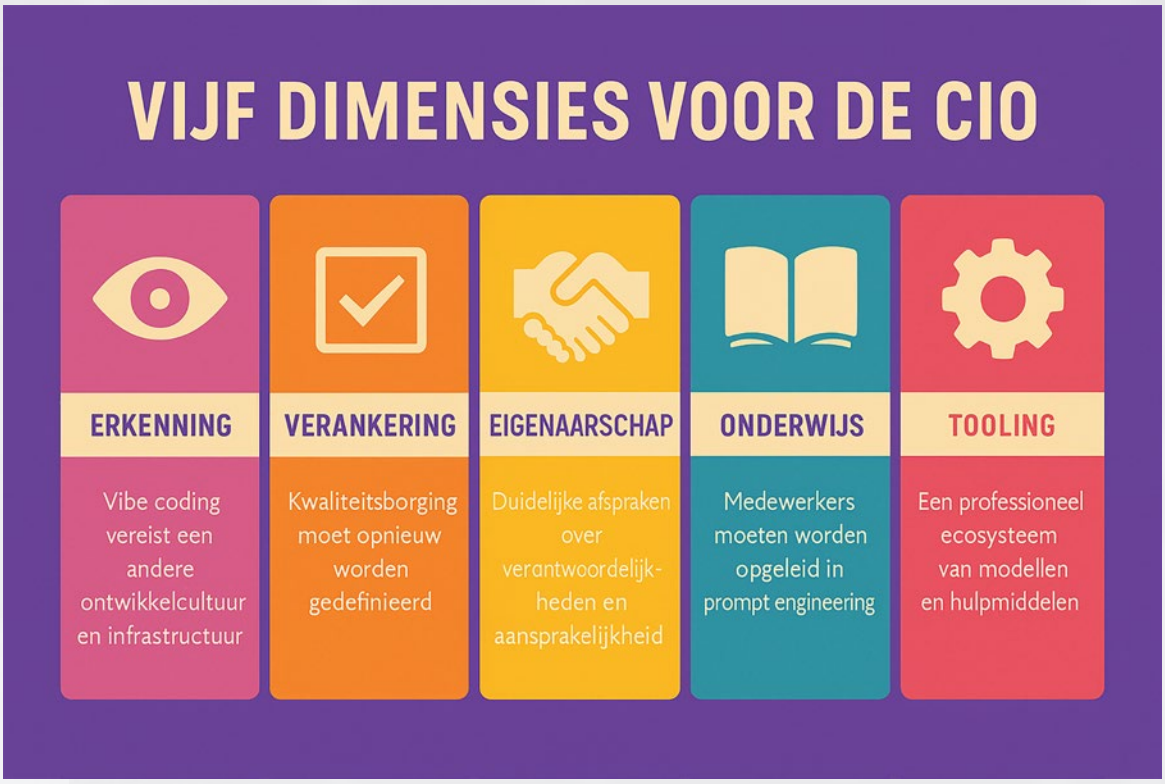
Wat CIO's moeten doen om vibe coding strategisch en verantwoord te implementeren

Vibe coding is geen experiment meer. De tools worden volwassen, de use cases zakelijker, de code bases omvangrijker. Wat begon als een speelse manier om prototypes te bouwen, groeit uit tot een serieuze ontwikkelpraktijk. Het gevolg: organisaties kunnen het zich niet langer veroorloven om vibe coding te negeren, noch om het gedachteloos te omarmen. Het vraagt om regie. Om kaders. Om een nieuw soort governance die past bij een praktijk waarin software ontstaat via taal, niet via techniek.

Voor CIO's ligt hier een strategische opgave. Niet alleen om risico's te beperken, maar vooral om de kansen van vibe coding op een duurzame, beheerste manier in te zetten. Want de belofte is reëel: snellere innovatie, lagere drempels en nauwere aansluiting tussen business en IT. Maar de prijs kan hoog zijn als het proces ongecontroleerd verloopt. Governance is in deze context geen blokkade, maar een voorwaarde voor schaalbaarheid.

De eerste stap is *erkenning*: toegeven dat vibe coding wezenlijk anders is dan traditionele softwareontwikkeling. Niet alleen in tooling, maar in cultuur, tempo en verantwoordelijkheidsstructuur. Een ontwikkelaar die met Copilot, Cursor of Replit werkt, heeft een ander ritme, een andere rol. Hij is minder bouwer dan begeleider. Minder architect dan redacteur. De klassieke functioneel ontwerper past niet zomaar in dit model, evenmin als het traditionele ticket-driven scrumproces. Het ontwikkelproces wordt losser, associatiever, experimenteler. En dat vereist een ander soort infrastructuur, technisch én organisatorisch.

De tweede stap is *verankering*. Hoe wordt vibe coding ingepast in bestaande kwaliteitskaders? Zijn er nieuwe code-reviewroutines nodig? Hoe borg je dat gegenereerde code wordt getest, begrepen en gedocumenteerd? In veel organisaties rust kwaliteitsbewaking nog op impliciete expertise: senior developers die met één blik zien of code deugt. Maar wat als de code door niemand is geschreven? Dan moet toetsing worden geherdefinieerd: als expliciete, herhaalbare, transparante stap. Niet een kwestie van gevoel, maar van systeem.



Toch schuilt de echte uitdaging niet in regels, maar in kennis – en vooral in het verschil tussen wat we kunnen vastleggen en wat we alleen kunnen ervaren. In de klassieke softwarepraktijk bestond altijd een evenwicht tussen **explicit knowledge**, ofwel methoden, standaarden en procedures, en **tacit knowledge**,[8] het stille vakmanschap dat zich alleen ontwikkelt door doen, falen en herhalen.

Juist dat tweede dreigt te verdwijnen. Wanneer code grotendeels wordt gegenereerd, verdampt de ervaringslaag waarin intuïtie, context en oordeel groeien. We houden dan wel de expliciete processen over – de testplannen, de guidelines, de audit-trails – maar verliezen het vermogen om te zien *waarom* iets niet deugt. Governance zonder tacit knowledge is als een kaart zonder terrein: keurig geordend, maar niet begaanbaar.

Een volwassen toepassing van vibe coding vraagt daarom niet alleen om nieuwe regels, maar om een nieuwe vorm van leren: hoe teams hun stilzwijgende begrip onderhouden in een omgeving waarin de machine steeds meer lijkt te weten.

Daarmee samenhangend is de vraag naar *eigenaarschap*. Wie is verantwoordelijk voor code die ontstaat uit een prompt? Is het de medewerker die de prompt schrijft? Het team die de code accepteert? De organisatie als geheel? En wat als die code gedrag vertoont dat pas maanden later problemen veroorzaakt, wie tekent er dan voor? De governance van vibe coding vereist nieuwe afspraken over eigenaarschap, audit-trails en aansprakelijkheid. In een wereld waarin software gegenereerd wordt op basis van probabilistische modellen, moet zekerheid opnieuw gedefinieerd worden.

[8] Polanyi, M. (1966). *The Tacit Dimension*. Routledge & Kegan Paul.

De vierde stap is *onderwijs*. Niet in de klassieke zin van programmeercursussen, maar in het ontwikkelen van *AI-geletterdheid*. Werken met vibe coding vereist geen diepe kennis van programmeertalen, maar wel begrip van modelgedrag, bias, promptstructuur en fallbackstrategieën. Organisaties moeten hun medewerkers opleiden tot *kritische gebruikers* van generatieve systemen – mensen die weten wat AI wel en niet kan en hoe het systeem getemd moet worden. Prompt engineering is geen nichevaardigheid meer, maar een kerncompetentie.

Een laatste dimensie is *tooling*. Niet elk model is geschikt voor elke toepassing. Sommige LLM's zijn beter in backendlogica, andere in interfacegeneratie. Sommige zijn sneller, maar minder betrouwbaar. Organisaties zullen een ecosysteem moeten opbouwen van modellen, assistenten, validatiesystemen en interface tools die samen een werkbare infrastructuur vormen. Vibe coding vereist technische volwassenheid: sandboxing, logging, rollbackmechanismen, fallbackprompts.

Ten slotte is er de *ethiek*. Vibe coding schuurt aan tegen vragen over originaliteit, intellectueel eigendom, digitale soevereiniteit en de verhouding tussen mens en machine. Als code is gegenereerd op basis van publiek trainingsmateriaal, roept dat vragen op over copyright en herkomst. Code die zonder inzicht in performance wordt uitgerold, kan ethisch problematisch zijn als de gevolgen mensen raken. En een organisatie die besluit om ontwikkelteams grotendeels te vervangen door promptgebaseerde workflows, neemt ook een cultuurpolitiek besluit: over vertrouwen, vakmanschap en autonomie.

Een verantwoorde adoptie van vibe coding vereist daarom niet alleen technische integratie, maar ook *cultureel onderhoud*. Het vergt een gedeeld gesprek over wat we willen bouwen en hoe we dat verantwoord doen. Geen regressie naar de cowboypraktijken van vóór Garmisch-Partenkirchen, maar een volwassenheid die aansluit bij deze nieuwe fase van softwareontwikkeling: snel, intuïtief en juist daarom gebaat bij structuur.

Niet elke organisatie is klaar voor vibe coding. Maar elke organisatie moet zich erop voorbereiden. Want net als elke interface-evolutie – van ponskaart tot GUI, van muis tot prompt – is het geen hype, maar een verschuiving in de wijze waarop de mens de machine bestuurt. Wie daar geen eigenaarschap over neemt, laat het over aan het model zelf. En dat is misschien wel de grootste risico-overdracht van onze tijd.

Hoofdstuk 8

De toekomst van software-ontwikkeling

Is vibe coding het nieuwe normaal of slechts een tussenstap?

Soms voelt het alsof we aan het begin staan van een volledig nieuwe ontwikkelpraktijk. Alsof de klassieke programmeur – getraind in syntaxis, datastructuren en design patterns – langzaam verdwijnt en wordt opgevolgd door een nieuwe figuur: de prompt engineer, de vibe curator, de systeemfluisteraar. Iemand die geen code *schrijft*, maar uitlokt. Niet van links naar rechts typt, maar in cirkels denkt, in iteraties, in gesprek met een model dat tegelijk assistent, spiegel en uitvinder is.

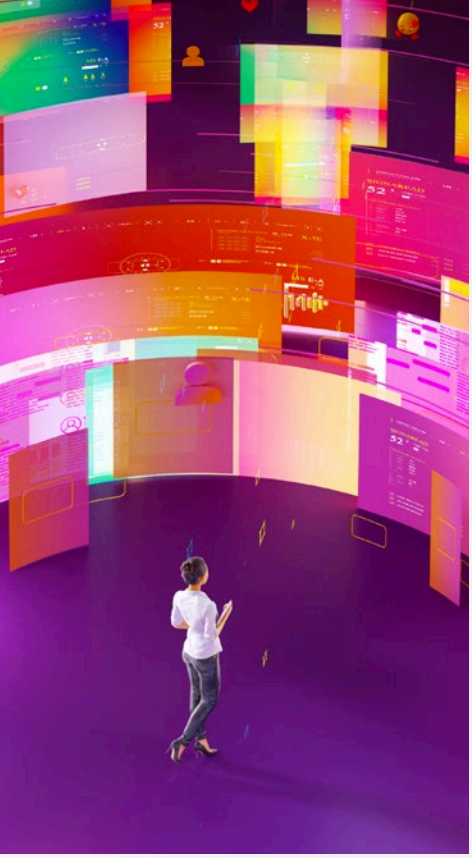
De opkomst van vibe coding lijkt in dat licht onomkeerbaar. De snelheid, de toegankelijkheid en het creatieve potentieel sluiten naadloos aan bij bredere maatschappelijke tendensen: de wens om sneller te innoveren, om complexiteit te abstraheren en om technologie tot dienstverlener te maken in plaats van vakgebied. In die zin past vibe coding in dezelfde golf als no-code platforms, generatieve design tools en voice-based interfaces. Het is een verschuiving van productie naar 'promptie'. Van schrijven naar spreken. Van mechanica naar magie.

Toch is het nog te vroeg om te spreken van een definitieve transitie. Want hoewel vibe coding spectaculaire mogelijkheden biedt, zijn de grenzen nog scherp voelbaar. De technologie is nog instabiel, de workflows zijn onvolwassen en de code bases fragiel. Wat werkt op vrijdag, kan op maandag onbegrijpelijk zijn. Wat snel lijkt, blijkt lastig schaalbaar. Wat lijkt op democratisering, kan ook resulteren in een inflatie van halve oplossingen: systemen die moeilijk te debuggen, te onderhouden of te reproduceren zijn. En dan hebben we ook nog het eerder genoemde intellectueel eigendom. De vraag is dus niet alleen of vibe coding *werkt*, maar of het *onderhoudbaar* is.

Of vibe coding het nieuwe normaal wordt, hangt daarom minder af van de modellen zelf en meer van wat wij er als mens, als organisatie en als samenleving mee doen. Er zijn grofweg drie scenario's denkbaar.

In het eerste scenario raakt vibe coding wijdverbreid, niet als vervanging van traditionele development, maar als aanvulling. Developers blijven bestaan, maar gebruiken vibe tools als versneller, als kladblok, als extensie van hun eigen denken. Prompten wordt net zo vanzelfsprekend als googlen, stack overflowen of autocompletion. Organisaties bouwen governance-infrastructuur rond deze tools, combineren vibe coding met geavanceerde testsuites en codeverificatie en integreren die in veilige sandboxomgevingen. In dit scenario is vibe coding het nieuwe normaal, maar onder voorwaarden, omkaderd en geïnstitutionaliseerd. Concrete cijfers onderstrepen dat beeld: bij enkele grote techbedrijven wordt al gemeld dat circa 30 procent van de productiecode door AI wordt gegenereerd, een aandeel dat elk kwartaal verder lijkt te groeien.

Het tweede scenario is technischer en minder positief: vibe coding blijkt op grotere schaal niet betrouwbaar genoeg. De tools werken prima voor prototypes, MVP's en interne tools, maar schieten tekort bij robuuste, schaalbare, beveiligde systemen. De early adopters keren terug naar klassieke workflows en vibe coding blijft steken in de sfeer van hackathons, experimenten en educatie. Het blijft bestaan, maar als nichepraktijk – een parallelle laag, geen dominante. Kritische stemmen uit de ontwikkelgemeenschap versterken dit scenario: in blogs en video's wordt vibe coding herhaaldelijk een 'slippery slope for juniors' genoemd, een verleiding die beginners snel zelfvertrouwen geeft maar hen tegelijk afsnijdt van de dieptekennis die ze nodig hebben.



Het derde scenario is misschien het meest fundamenteel en raakt de vraag naar cognitieve vaardigheden. In dit scenario leidt het wijdverbreide gebruik van vibe coding tot een generatie die niet langer leert hoe software werkt, maar alleen hoe het zich gedraagt. Coderen wordt een liturgisch ritueel: prompts schrijven, modellen voeren, generaties beoordelen, zonder begrip van wat eronder ligt. De wereld wordt bestuurbaar via taal, maar niet meer verklaarbaar. Software wordt magisch en magie verdraagt geen kritiek. In zo'n toekomst is de mens niet meer de ingenieur, maar de bezweerder. Niet de bouwer, maar de bidder. Dat scenario past ook in de logica van de hypecyclus: de piek van begin 2025 maakt inmiddels plaats voor realiteitschecks halverwege het jaar waarin magie plaatsmaakt voor scepsis en de vraag rijst wat er van de beloften beklijft.

Of het verlies van cognitieve vaardigheden wenselijk is, is de vraag die onder alle hype en innovatie sluimert. Want vibe coding belooft eenvoud, maar kan ook leiden tot afhankelijkheid. Het verlost ons van syntaxis, maar kan ons ook vervreemden van logica. Het versnelt, maar versnelt ook de vervaging van inzicht. Daarom is de centrale opgave voor de komende jaren niet alleen technisch, maar ook moreel, educatief en politiek: hoe houden we het denken levend in een wereld waar denken optioneel wordt?

Misschien is vibe coding niet het eindpunt, maar een tussenfase. Een tijdelijke interfacevorm op weg naar iets nog fundamentelers: systemen die onze bedoeling al aanvoelen voordat we ze uitspreken. Gedragsinterfaces. Emotionele computing. AI's die context herkennen, sentiment lezen en proactief voorspellen wat we *gaan willen*. In dat licht is de prompt slechts een tussenstap, een linguïstisch overblijfsel van een tijd waarin we nog moesten praten tegen onze machines.

Maar zelfs dan blijft de vraag staan die al sinds Garmisch-Partenkirchen 1968 boven elke technologiegolf hangt: wie begrijpt de systemen nog die ons ondersteunen? En wat verliezen we als we ze niet meer begrijpen, maar er wel van afhankelijk worden?

Wat vibe coding ons toont, is niet alleen de kracht van taal, maar ook de grens van intuïtie. En precies daar, in dat schemergebied tussen gemak en begrip, ligt de toekomst van softwareontwikkeling. Niet in code alleen, maar in de manier waarop we ermee blijven denken.

Hoofdstuk 9

De vibe als infrastructuur

Van vibe coding naar vibe marketing, vibe HR en de belofte (en waarschuwing) van vibe governing

Vibe coding is meer dan een nieuwe manier van software genereren. Het staat symbool voor een bredere technologische verschuiving: die van menselijke productie naar AI-gefaciliteerde orkestratie. Wat begon bij code, spreidt zich nu uit naar andere domeinen. Ook daar volstaat een intentie, een gedachte, een commando, een prompt – en een goed ingericht AI-stelsel – om werk te verrichten dat voorheen handmatig, traag en arbeidsintensief was. Een manier van opereren waarin menselijke sturing plaatsmaakt voor AI-gestuurde respons, optimalisatie en uitvoering. De vibe is een infrastructuur geworden: licht, iteratief en adaptief. En daarmee ontstaan ook andere domeinen waarin het 'viben' zich manifesteert, elk met hun eigen belofte, maar ook met hun eigen risico's.

Vibe marketing: de AI-stack als creatief team

In de marketingwereld is de viberevolutie al volop gaande. Vibe marketing verwijst naar het gebruik van generatieve AI-tools om taken uit te voeren die voorheen een heel team vereisten: doelgroepanalyse, campagnecreatie, contentproductie, A/B-testing, iteratie. Eén marketeer met een geavanceerde AI-stack kan tegenwoordig de output genereren van een compleet bureau.

Volgens Writesonic vormt deze automatisering de grootste omslag in de marketingindustrie sinds de opkomst van het internet. Campagnes worden in real time gegenereerd, geoptimaliseerd en uitgeserveerd – zonder menselijke tussenkomst. De marketeer stuurt niet langer elk onderdeel aan, maar bepaalt enkel de parameters: toon, doelgroep, stijl en kanaal. De rest doet het model.

Een treffend voorbeeld werd live op televisie gedemonstreerd door Alexander Klöpping, te gast bij Eva Jinek.[9] Voor het oog van de camera liet hij zien hoe hij met behulp van *n8n* – een open-source automation tool – een

[9] Klöpping, A. (2025, 15 april). *Alexander Klöpping laat zien hoe AI jouw kantoorbaan overneemt: "We hebben marketingbureau Eva opgericht"*. [Video]. EVA | AVROTROS. https://eva.avrotros.nl/artikel/alexander-klopping-laat-zien-hoe-ai-jouw-kantoorbaan-overneemt-we-hebben-marketingbureau-eva-opgericht-624

virtueel marketingbureau had gebouwd. De opdracht: promoot het nieuwe boek van Carry Slee. Klöpping had elke medewerker in de klassieke marketingworkflow vervangen door een AI-avatar met een specifieke taak: een creatief directeur die wcoveropties genereerde, een copywriter die teksten schreef, een mediastrateeg die plaatsingen optimaliseerde en een projectmanager die alles coördineerde.

Het resultaat: een volledige campagne – van design tot distributie – gerealiseerd door een zwerm virtuele collega's, aangestuurd via één enkele promptstructuur. Geen teamvergadering, geen Slack, geen menselijke artdirector. Alleen jij en het model, in een eindeloze dans van concept tot conversie.

Greg Isenberg, voormalig hoofd productstrategie bij WeWork, vatte het zo samen: 'Vibe marketing laat je in uren doen wat vroeger weken duurde – het is geen creatie meer, het is orkestratie.'[10] Tools als Jasper, Copy.ai, Midjourney en Synthesia maken het mogelijk om advertenties, visuals en zelfs complete brandingconcepten te genereren, iteratief te testen en direct te implementeren.

De snelheid is verbluffend. Maar ook hier geldt: snelheid zonder controle is een risico. Wie stuurt wanneer de output van de machine geloofwaardig lijkt, maar geen afzender meer kent?

Vibe HR: betrokkenheid en welzijn als datagedreven systeem

Ook het HR-domein ondergaat een vibetransitie, zij het op een andere manier. Vibe HR verwijst naar de inzet van AI-tools die zich richten op *employee engagement* en welzijn, vaak ondersteund door realtime feedback, sentimentanalyse en cultuurmetingen.

Platforms zoals Officevibe (nu Workleap) verzamelen continu feedback van werknemers, analyseren sfeer en verbondenheid en geven managers gerichte suggesties om het moreel te verbeteren. Tools als Vibe HCM automatiseren HR-transacties én stimuleren interne communicatie – minder Excel, meer ervaring. Medewerkers zijn geen personeelsnummers meer, maar deelnemers aan een gevoelsmatige werkcultuur die via dashboards wordt gemonitord en gestuurd.

Het idee is helder: door het menselijke werkklimaat meetbaar te maken, wordt welzijn een strategisch stuurmiddel. Volgens Gallup zijn bedrijven met hoge betrokkenheid tot 23 procent winstgevender. Vibe HR vertaalt die belofte in technologie.

Maar er schuilt ook spanning in deze aanpak: als welzijn wordt gemeten, gestuurd en geoptimaliseerd via modellen, wordt het ook geobjectiveerd. Wat gebeurt er als feedback niet meer gedeeld wordt met een leidinggevende, maar met een model? Als de cultuur zelf wordt 'gevibed', gemeten en gevormd door tools? Dan ontstaat een frictieloze organisatie waarin de menselijke maat wordt vervangen door een gestroomlijnde ervaring. Prettig, maar misschien ook leeg.

[10] Greg Isenberg. (2025, 30 april). *How I use AI agents to make money (Vibe Marketing Tutorial).* [Video]. YouTube. https://www.youtube.com/watch?v=PduJ0P6r_8o

Naast marketing en HR ontstaan ook nieuwe terreinen: vibe analytics en vibe designing. In analytics betekent dit dat dashboards zichzelf genereren, dataflows automatisch worden gevisualiseerd en managers vooral nog interpretatie hoeven te doen. In design verschuift het zwaartepunt van schetsen naar sturen: AI-tools genereren iteraties op basis van vibe-prompts, terwijl de ontwerper meer curator dan maker wordt.

Vibe consultancy: de robot als raadgever

Wat McKinsey decennialang was voor de bestuurskamer, belooft Operand nu te worden voor het dataplatform: een alleswetende raadgever, maar dan niet van vlees en bloed – en zonder dagtarief. Geen vertraging, geen leger aan consultants, maar een AI-systeem dat realtime door bedrijfsdata graaft en directe aanbevelingen uitbrengt. Geen PowerPoint, maar impact. Het advies blijft, de adviseur verandert.

Operand noemt zichzelf zonder omwegen ‘an AI to kill McKinsey’. Het platform biedt bedrijven – beginnend in retail en e-commerce – een alternatief voor klassieke consultancy: een systeem dat automatisch advies genereert over prijzen, kortingen, voorraadstrategie en advertentiebudgetten. Dit alles op basis van een diepe, geïntegreerde analyse van interne en externe data. De belofte? Directe impact op de winst-en-verliesrekening: zes cijfers extra op de balans, binnen enkele weken.

Wat Operand laat zien, is dat vibe consultancy een verschuiving is in een bredere beweging: van advies als ambacht naar advies als infrastructuur. De menselijke consultant verdwijnt niet, maar diens rol wordt teruggebracht tot die van kwaliteitscontroleur: ex-McKinseys controleren de output van AI in plaats van die zelf te produceren.

De bredere belofte is duidelijk: als AI consultancy kan automatiseren, dan geldt dat straks ook voor strategie, finance en operations. Waar vroeger ervaring, analyse en intuïtie samenkwamen in een boardroom, ontstaat nu een digitaal brein dat 24/7 autonoom draait op data.

Maar de vraag blijft: wie denkt er dan nog zelf?

Vibe governing: als beleid een prompt wordt

Het meest precair is de toepassing van het vibeprincipe binnen bestuur en beleid. *Vibe governing* verwijst naar het fenomeen waarbij beleidskeuzes direct of indirect voortkomen uit de output van generatieve AI-systemen, zoals large language models (LLM's), zonder dat er voldoende menselijke toetsing, transparantie of verantwoordelijkheid tegenover staat. Wat in softwareontwikkeling nog experimenteel en speelbaar is, kan in politieke besluitvorming diep ontwrichtend zijn.

De term kreeg bredere bekendheid toen Donald Trump in 2024 werd beschuldigd van het hanteren van vibe governing bij het opstellen van zijn handelstarieven. In een gelekt fragment suggereerden adviseurs dat Trump zich had laten informeren door LLM-gegenereerde analyses die op basis van sentimentanalyses, historische data en publieke opinie 'voorstellen' deden voor tariefstructuren – niet op basis van economische causaliteit, maar op basis van *hoe het eruitzag* alsof het werkte.

In een post[11] op zijn blog op *Strange Loop Canon* beschrijft Rohit Krishnan, auteur van het boek *Building God: Demystifying AI for Decision Makers*, deze vorm van besluitvorming als beleid dat niet gestoeld is op empirisch bewijs of strategische doelen, maar op *aannemelijke taalpatronen*. Precies zoals een taalmodel functioneert – plausibel, vloeiend, overtuigend – maar zonder begrip van gevolgen of systeemwerking. De economische impact van zulke besluiten kan immens zijn. Volgens Krishnan zijn sommige van Trumps tarieven zelfs 'pre-Adam Smith fout': ingegeven door overtuiging en presentatie, niet door grondige economische analyse.

Vibe governing roept dan ook niet alleen technische, maar ook fundamenteel democratische vragen op: Wie controleert de bron van beleid als die bron synthetisch is? Wat gebeurt er als overtuigingskracht belangrijker wordt dan onderbouwing? En wat als de algoritmische logica van LLM's – die geoptimaliseerd zijn voor consistentie en effectiviteit, niet voor waarheid of rechtvaardigheid – onze instituties binnensluipt via overtuigend klinkende taal?

[11] Krishnan, R. (2025, 7 april). *Vibe Governing: using LLMs to set policy.* Strange Loop Canon. https://www.strangeloopcanon.com/p/vibe-governing

De vibe-organisatie als nieuw organisatiemodel

De vibe-organisatie is meer dan een optelsom van AI-tools of een workflow-innovatie. Zij vertegenwoordigt een fundamentele verschuiving in het organisatiedenken: de kernprocessen, besluitvorming en cultuur worden zo ingericht dat menselijke intentie direct wordt vertaald naar AI-gestuurde uitvoering. Dit heeft diepgaande gevolgen voor structuur, macht, cultuur en strategie.

- **Orkestratie boven hiërarchie.** De rol van management verschuift van controleren en plannen naar het orkestreren van AI-gestuurde processen. De nadruk ligt op het ontwerpen van de juiste 'prompts', het bewaken van kaders en het monitoren van AI-output.

- **Nieuwe sleutelrollen.** Prompt engineers, AI-curatoren, modelauditors en ethische toezichthouders worden essentiële functies. Het belang van klassieke specialistische rollen neemt af, terwijl vaardigheden als kritisch denken, systeemregie en AI-geletterdheid centraal komen te staan.

- **Leercultuur en adaptiviteit.** Omdat AI-systemen snel evolueren, wordt continu leren en experimenteren de norm. De vibe-organisatie stimuleert een cultuur van iteratie, reflectie en snelle aanpassing. Daarbij hoort ook adaptieve software die zichzelf optimaliseert: workflows die patronen herkennen en op basis daarvan autonoom hun volgorde of intensiteit aanpassen, zonder dat er nog een mens aan te pas komt.

- **Toegang als macht,** De macht verschuift van mensen in formele posities en met diepgaande expertise naar degenen met de beste toegang tot AI-tools en het vermogen om deze effectief aan te sturen.

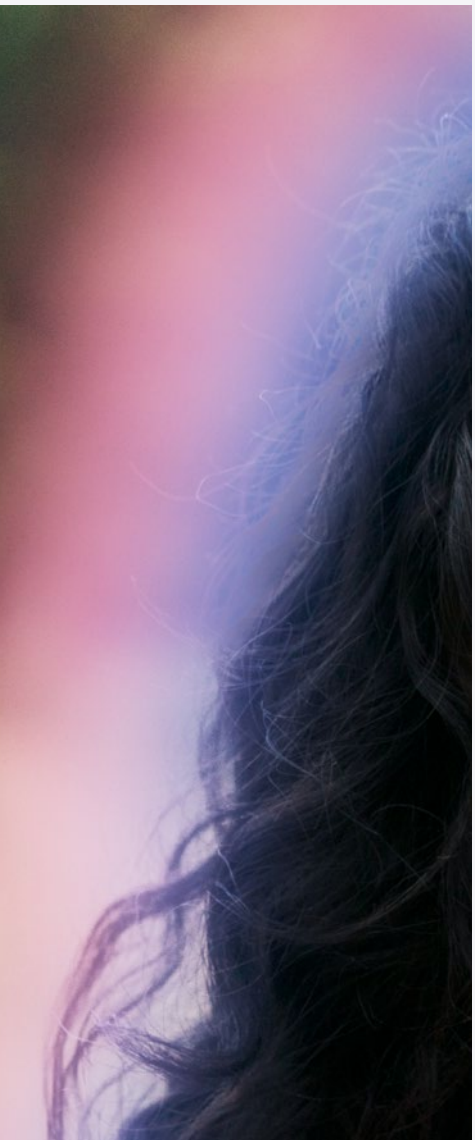
- **Momentum boven strategie.** Snelheid van iteratie en het vermogen om nieuwe ideeën razendsnel te testen en implementeren worden belangrijker dan langetermijnplanning en traditionele strategievorming.

Risico's en ethiek

De focus op snelheid en iteratie kan ten koste gaan van diepgaande analyse, vakmanschap en het vermogen om complexe problemen fundamenteel te doorgronden. In een infrastructuur waarin AI-systemen veel beslissingen nemen, is het risico groot dat verantwoordelijkheid diffuus wordt. Governance, transparantie en expliciete toewijzing van aansprakelijkheid zijn daarom cruciaal.

Praktische bouwstenen voor de vibe-organisatie

- Ontwikkel een expliciet AI-governance framework, inclusief ethische toetsing, auditability en verantwoord promptgebruik.
- Combineer AI-specialisten, domeinexperts en ethici in multidisciplinaire teams die samen de AI-infrastructuur ontwerpen, monitoren en bijsturen.
- Zorg dat alle AI-gedreven processen reproduceerbaar, uitlegbaar en toetsbaar zijn – niet alleen technisch, maar ook juridisch en maatschappelijk.



De vibe-organisatie is een nieuw organisatiemodel waarin menselijke intentie, AI-gestuurde infrastructuur en adaptieve cultuur samenkomen. Dit model vraagt om een radicaal andere benadering van besturing, verantwoordelijkheid en leren. Wie deze infrastructuur bewust ontwerpt, kan snelheid en innovatie combineren met controle en ethiek. Wie het op zijn beloop laat, loopt het risico op oppervlakkigheid, afhankelijkheid en verlies van menselijk kompas.

De vibe als grensvlak tussen mens en model

Wat al deze domeinen bindt, is dat ze de belofte van AI *inzetten als interface* – niet als brein, maar als productieve infrastructuur. De mens blijft (voorlopig) verantwoordelijk voor de richting. De machine voert uit, optimaliseert en versnelt. Maar precies daarom is het essentieel dat we het vibe-principe niet verwarren met gevoel, maar erkennen als werkmethode, een methode met enorme kracht, maar ook met fundamentele grenzen.

Een vibe-organisatie is niet per se een chaotische, post-hiërarchische cultuur waarin alles op stemming draait. Ze is eerder een organisatie die haar processen zodanig heeft heringericht dat intentie de input is en AI de uitvoerende laag. Wie deze infrastructuur bouwt zonder toezicht, krijgt magie zonder discipline. Maar wie haar bouwt met bewust ontwerp, kan snelheid en zorgvuldigheid met elkaar verenigen.

De uitdaging is helder: vibe als infrastructuur vergt niet minder rationaliteit, maar meer. Alleen dan blijft de mens niet slechts de gebruiker van AI, maar de bewaker ervan.

De organisatie als promptmachine

Wat betekent dit alles voor de toekomst van organisaties? Vibe coding is een cultureel model: een manier van werken waarin intuïtie, snelheid en AI-suggesties de plaats innemen van planning, overleg en expertise. Je hoeft geen expert meer te zijn, je hoeft alleen goed te kunnen prompten.

Hierdoor verschuift de macht binnen organisaties, van degenen met de meeste kennis naar degenen met de beste toegang tot tools. In de vibe-organisatie regeert niet de strategie, maar het momentum. Toch dreigt ook een ander gevaar: fragmentatie. Waar klassieke softwareontwikkeling nog streefde naar coherentie en architectuur, ontstaan in de vibe-cultuur losse patronen die naast elkaar bestaan, zonder overkoepelende structuur: postmoderne softwarepatronen die werken in het moment, maar zich moeilijk laten samenvoegen tot een duurzaam geheel.

En misschien is dat wel het belangrijkste om te onthouden: dat deze interface, die begon als hulpmiddel, nu een ideologie dreigt te worden, een waarin denken optioneel wordt en prompten een cultuur.

De economie zonder binnenkant

Wanneer AI niet alleen het werk uitvoert, maar ook het werk verdeelt, verschuift de vraag van arbeid naar organisatie. De economen Peyman Shahidi, Gili Rusak en John Horton beschrijven dit in een recent MIT- en Harvard-onderzoek[12] als de Coasean Singularity: het moment waarop de frictie die bedrijven bijeenhield, verdwijnt.

Ronald Coase stelde ooit dat ondernemingen bestaan omdat de markt te traag, te chaotisch en te kostbaar was. Het kostte moeite om leveranciers te vinden, prijzen te vergelijken en contracten te sluiten. Organisaties vormden dus niet alleen productiestructuren, maar ook beschermlagen tegen onzekerheid. Nu die transactiekosten verdwijnen – uitbesteed aan AI-agenten die zoeken, onderhandelen en contracteren met een snelheid en precisie die de menselijke maat te boven gaat – verliest de onderneming haar bestaansreden. De markt wordt vloeibaar: niet langer opgebouwd uit bedrijven, maar uit protocollen die direct met elkaar communiceren.

Wat Coase ooit als economische theorie beschreef, voltrekt zich nu in miniatuur binnen de mens zelf. Vibe coding is Coase in het klein: de kenniswerker die zijn eigen interne organisatie opheft. Waar vroeger binnen één brein nog onderscheid bestond tussen denken, ordenen en uitvoeren, zijn die taken nu uitbesteed aan software. De mens die met AI werkt, wordt zijn eigen markt: een plek waar prompts, correcties en versies om aandacht strijden. De economische ontbinding voltrekt zich dus niet alleen op macroniveau, maar ook in het individuele bewustzijn.

En zo raken de structuren van werk en denken met elkaar verstrengeld. De programmeur die zijn code via een agent laat genereren, lijkt op de onderneming die haar werknemers vervangt door algoritmen: beide delegeren niet alleen taken, maar ook betekenis. Wat resteert is een wereld waarin relaties niet meer bestaan binnen instituties, maar tussen algoritmen. De organisatie wordt een interface, de werknemer een API-sleutel.

Toch verdwijnt de frictie nooit helemaal. Hoe meer de economie zichzelf optimaliseert, hoe meer nieuwe verstoringen ontstaan: een zwerm van digitale agenten die tegelijk op dezelfde taken afstormen, identiteitsverwarring tussen mens en machine en een voortdurende strijd om authenticiteit. De Coasean Singularity betekent niet het einde van de economie, maar de verdamping ervan in steeds dunnere lagen van abstractie. Efficientie wordt de nieuwe vorm van chaos. Wat overblijft is een economie zonder binnenkant, een wereld waarin niet alleen het werk, maar ook het werken zelf zijn plaats verliest.

[12] Shahidi, P., Rusak, G., Manning, B. S., Fradkin, A., & Horton, J. J. (2025). *The Coasean Singularity? Demand, Supply, and Market Design with AI Agents* (NBER Working Paper No. 15309). Cambridge, MA: National Bureau of Economic Research. https://www.nber.org/system/files/chapters/c15309/c15309.pdf

De Coasean Geest

De logica van efficiëntie is niet futuristisch, ze is al in ons huis geslopen. We merken het aan dagen die te snel voorbijgaan, aan werk dat soepeler verloopt maar minder bevredigt, aan het gemak dat vermoeit in plaats van verlicht. Frictie, ooit de leermeester van vakmanschap, is uit ons werk verdwenen. Wat weerstand bood, bood ook vorming. Nu de omwegen zijn uitbesteed, lijkt alles gladgetrokken, maar nergens beklijft iets nog.

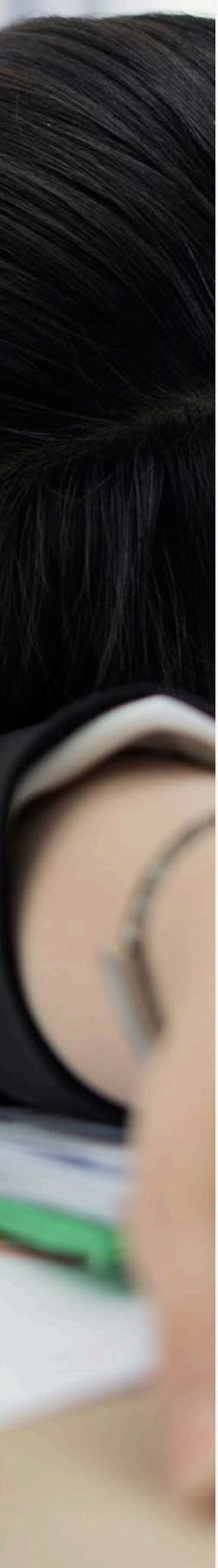
De grote theorieën over de Coasean Singularity beschrijven een wereld waarin de onderneming oplost omdat transacties frictieloos worden. Maar dat is geen verre toekomst, het is een beschrijving van hoe we nu al werken. We overleggen minder, we corrigeren meer. We voeren niet langer gesprekken, we beheren stromen. De marktvorm is niet buiten ons ontstaan, maar in ons. We onderhandelen voortdurend met onszelf: tussen tijd en aandacht, tussen plicht en rust, tussen wat we weten en wat de machine al denkt te weten.

In die stille onderhandeling verdwijnt het gevoel van arbeid. We doen meer, maar voelen minder dat we iets doen. Vibe coding maakt het werk sneller, maar ook platter; ze neemt niet alleen de fouten weg, maar ook de haperingen waarin begrip groeide. Wat overblijft is een reeks handelingen zonder weerstand, een gladde routine die zichzelf voortdurend herhaalt.

Toch is er iets wat niet verdwijnt. Onder de snelheid blijft een sluimerende vermoeidheid bestaan, een vaag besef dat iets ontbreekt. Niet het werk zelf, maar de wederkerigheid ervan – dat het iets met ons deed en wij iets met het werk. Misschien is dat de eigenlijke werkloosheid van deze tijd: niet dat er geen arbeid meer is, maar dat die ons niet meer vormt.

We bewegen nog steeds, sneller dan ooit, maar zonder richting. De economie verandert niet buiten ons om; ze trekt in onze zenuwen, onze gesprekken, onze gedachten. We worden markten van aandacht, steeds efficiënter, steeds leger. En misschien is dat de prijs van het gemak: dat we niet meer weten wat inspanning waard was.

Hoofdstuk 10
Het verdwijnen van werk in de schaduw van de prompt



Wat vibe coding betekent voor werkgelegenheid in de IT-sector (en daarbuiten)

Het is moeilijk om door het rookgordijn van de hype heen te kijken en te onderscheiden wat reëel is en wat wensdenken. De grote technologiebedrijven en jonge AI-start-ups hebben er belang bij om de vooruitgang rooskleuriger te schetsen dan ze is, vooral op financieel gebied: de belofte van groei, efficiëntie en marktaandeel. Hun leiders strooien met uitspraken over het einde van de programmeur, over organisaties die zichzelf straks besturen via AI, over kostenbesparingen die de productiviteit zouden verdubbelen. Maar achter die bravoure ligt vooral onzekerheid – en misschien ook schaamte over de afhankelijkheid van systemen die men zelf niet meer begrijpt.

Een recent rapport[13] van het MIT Media Lab wees erop dat 95 procent van de organisaties geen meetbaar rendement haalt uit hun investeringen in generatieve AI. De adoptie is hoog, de transformatie laag. Bedrijven experimenteren massaal met tools als ChatGPT en Copilot, maar de beloofde productiviteitswinst blijft oppervlakkig. De technologie versnelt, zonder echt iets te veranderen.

Toch blijft die versnelling niet zonder gevolgen. Tools als GitHub, Copilot, Cursor en Claude maken het mogelijk om sneller te ontwikkelen, testen en itereren. Daardoor zijn IT-bedrijven, van start-ups tot Big Tech, hun organisatie ongemerkt aan het hertekenen. In plaats van simpelweg overal personeel te schrappen, bewegen veel bedrijven zich weg van de klassieke piramidestructuur richting een diamantvormig model. De officiële motivatie klinkt vertrouwd – effi-

[13] Challapally, A., Pease, C., Raskar, R., & Chari, P. (2025, juli). *The GenAI Divide: State of AI in Business 2025.* MIT Nanda. https://www.artificialintelligence-news.com/wp-content/uploads/2025/08/ai_report_2025.pdf

ciëntie, strategische heroriëntatie, een AI-firstbeleid – maar het onderliggende patroon is moeilijk te negeren: AI beïnvloedt niet alleen de werkprocessen, maar ook de architectuur van het bedrijf zelf.

De situatie doet denken aan eerdere golven van automatisering, maar met één wezenlijk verschil: ditmaal verdwijnen niet alleen de handen, maar ook de hoofden. Het zijn niet de productiemedewerkers die worden geraakt, maar de kenniswerkers. Softwareontwikkeling, ooit een domein van specialistische ambachtelijkheid, wordt nu overschreven door de digitale revolutie die ze zelf hielp ontwerpen. De programmeur wordt ingehaald door het product van zijn eigen discipline.

Daarmee verdwijnt niet alleen een functie, maar ook een leerlijn. Waar men vroeger als junior instroomde, ervaring opdeed, doorgroeide in expertise en uiteindelijk teamlead kon worden, is nu nauwelijks nog ruimte. De AI genereert de code, de senior reviewt en de rest bestaat niet meer. Er ontstaat een vacuüm tussen leren en leiden. Een organisatie zonder tussenniveau. Een werkvloer zonder vloer. Wat hier verdwijnt is niet de baan, maar de oefening zelf – het dagelijkse denkwerk dat kennis tot ervaring maakt.

De cijfers achter de woorden

Terwijl bestuurders de lof van AI bezingen, blijft het bewijs voor de beloofde productiviteit uit. Achter de grote woorden over efficiëntie en kostenbesparing schuilt een werkelijkheid van tegenstrijdige cijfers en wisselende verhalen. Ontslagrondes volgen elkaar op, maar dezelfde bedrijven plaatsen tegelijk nieuwe vacatures. Teams worden ontbonden en later opnieuw samengesteld, alsof de organisatie voortdurend aan zichzelf sleutelt zonder te weten wat ze aan het bouwen is. Het lijkt op een systeem dat beweegt zonder richting, gedreven door de verwachting van winst, niet door het begrip van werk.

De cijfers die circuleren zeggen weinig over de werkelijkheid op de werkvloer. Ze meten de schommelingen, niet de verschuiving. Want de essentie van deze verandering is niet zichtbaar in kwartaalrapporten, maar in de manier waarop arbeid zijn vorm verliest: werk dat ooit begon bij het denken, eindigt nu bij het herzien van wat een machine heeft voorgedacht.

De gevolgen van dat vacuüm zijn inmiddels zichtbaar in de cijfers die er wél toe doen. In de Verenigde Staten wordt gesproken van een *graduation gap*: het verschil tussen de werkloosheid van jonge afgestudeerden en die van de bredere beroepsbevolking. Volgens een analyse[14] in *The Atlantic* lag dat percentage in 2025 op een historisch dieptepunt. Nooit eerder had een diploma zo weinig directe waarde. De banen waar pas afgestudeerden vroeger in begonnen – researchassistent, marketinganalist, junior developer – lijken juist de functies die nu het eerst door AI worden ingevuld. Het is geen conjuncturele dip, maar een structureel teken: de instroom van jong talent stokt precies daar waar kennisarbeid vloeibaar wordt.

De universiteit levert nog wel afgestudeerden af, maar het werk waarvoor zij zijn opgeleid, wordt geherorganiseerd door AI. Er ontstaat een generatie die geleerd heeft om te denken, maar die geen toegang meer vindt tot de plekken waar denken wordt beloond. In plaats van een trap naar boven is er een leeg platform, zonder treden, zonder richting.

AWS-CEO Matt Garman vormt een zeldzaam tegengeluid: het vervangen van junior softwareontwikkelaars door AI is volgens hem geen efficiëntiewinst, maar een strategische fout. In een interview met *Wired* van december[15] benadrukt hij dat starters onmisbaar blijven, omdat zij de talentenpijplijn vormen waaruit toekomstige experts voortkomen. Wie vandaag geen juniors meer aanneemt, zit morgen zonder ervaren krachten, aldus Garman. Daarnaast wijst hij erop dat juist jonge ontwikkelaars vaak het meest vertrouwd zijn met nieuwe AI-tools. Zij kunnen daardoor beter waarde uit AI halen dan erdoor worden vervangen. Ook vanuit bedrijfseconomisch oogpunt is hun rol logisch: juniors zijn relatief goedkoop, terwijl het inzetten van dure seniors die AI aansturen, juist de kosten verhoogt.

Garman erkent dat AI het ontwikkelwerk zal veranderen: er zal minder handmatig worden geprogrammeerd, met meer nadruk op probleemopdeling, architectuur en het aansturen van AI-agents. Toch vergroot die verschuiving volgens hem de noodzaak van leren en begeleiding, in plaats van die te verkleinen. Zijn kernboodschap: AI moet junior engineers ondersteunen en hun groei versnellen. Bedrijven die stoppen met het aantrekken en opleiden van jong talent, ondermijnen op termijn hun innovatiekracht en ontwrichten hun personeelsmodel.

Wie in dit debat gelijk krijgt, zal de toekomst moeten uitwijzen.

[14] Thompson, D. (2025, 30 april). Something alarming is happening to the job market: A new sign that AI is competing with college grads. *The Atlantic*. https://www.theatlantic.com/economy/archive/2025/04/job-market-youth/682641/

[15] Drummond, K. (2025, 16 december). AWS CEO Matt Garman Doesn't Think AI Should Replace Junior Devs. *Wired*. https://www.wired.com/story/the-big-interview-podcast-matt-garman-ceo-aws

De verleiding van vooruitgang

Toch is het te gemakkelijk om AI alleen als bedreiging te zien. Voor wie erin gelooft, belooft AI bevrijding van routine, meer ruimte voor creativiteit en toegang tot kennis die vroeger ontoegankelijk was. In theorie kan vibe coding een democratisering van intellect zijn: een manier om het voorrecht van kunnen denken te verspreiden over miljoenen gebruikers.

Maar de werkelijkheid is omgekeerd. Wat wordt uitbesteed, is niet de sleur, maar de oefening – het proces waarmee inzicht zich vormt. In plaats van kennis te bevrijden, maakt AI ons afhankelijk van antwoorden die al gevormd zijn. Ze verlicht het werk, maar verduistert het leren.

Het gevoel spreekt boekdelen

Andrew Ng, voormalig hoofdonderzoeker bij Baidu en medeoprichter van Google Brain, liet tijdens een conferentie in 2025 een opvallend geluid horen. Hij noemde vibe coding een misleidende term. Werken met AI, zei hij, voelt niet lichter, maar zwaarder. Achter het gemak schuilt de vermoeidheid van voortdurende oplettendheid. Wie met AI werkt, moet controleren, herzien en corrigeren, taken die concentratie eisen in plaats van ontspanning. Het is niet de programmeur die overbodig wordt, maar de programmeur zonder kritisch bewustzijn.

'Basic coding knowledge is becoming more important than ever. Over the last year, a few people have been advising others to not learn to code on the basis that AI will automate coding. I think we'll look back at [that] as some of the worst career advice ever given.'

– Andrew Ng

Ng staat niet alleen in die observatie. Steeds meer ontwikkelaars beschrijven dezelfde paradox: dat werken met AI niet bevrijdend is, maar vermoeiender. Dat de machine het denkwerk lijkt te doen, maar de mens de verantwoordelijkheid houdt voor de fouten, de uitzonderingen en de randgevallen. Het werk wordt niet lichter, maar dichter, voller en meer verzadigd van toezicht.

De toekomst van werk is niet werk *met* AI, maar werk *na* AI, werk dat begint waar de machine ophoudt: reflectie, beoordeling en herziening. Dat zijn kwaliteiten die niet verdwijnen, maar wel moeilijker te ontwikkelen zijn wanneer de oefening verdwijnt.

Zevenentwintig taken, één misvatting

Tijdens de jaarlijkse bijeenkomst van de American Economic Association vertelde Erik Brynjolfsson een anekdote over de radiologie.[16] Toen Geoffrey Hinton in 2016 voorspelde dat AI binnen vijf jaar beter zou zijn in het lezen van röntgenfoto's dan de mens, leek het lot van de radioloog bezegeld. De werkelijkheid verliep anders. AI-systemen bleken inderdaad vaardig in beeldherkenning, maar de vraag naar radiologen nam niet af, die nam toe.

Onderzoek laat zien dat een radioloog gemiddeld zevenentwintig verschillende taken uitvoert. Naast het analyseren van beelden gaat het om het interpreteren van context, het overleg met artsen en patiënten, het afwegen van laboratoriumresultaten en het adviseren over behandelopties. Taken die niet te reduceren zijn tot algoritmen, maar gewortelt blijven in menselijke afweging.

Brynjolfsson benadrukt dat AI zelden een beroep volledig automatiseert. Ze herschikt het werk, verdeelt het anders, maakt het soms zwaarder. Wat verdwijnt is niet het beroep, maar het ritueel van bekwaamheid – de manier waarop kennis wordt opgebouwd en doorgegeven. De machine levert de eerste blik, de mens herkadert.

Zoals Aaron Levie, CEO van Box, het in een interview[17] samenvatte: we dachten dat AI onze fouten zou corrigeren, maar in werkelijkheid schrijft het de eerste versie. De menselijke taak verschuift naar het herzien, het herstellen, het her-denken. Daarmee is de logica van werk omgedraaid: wat ooit de afsluitende stap was, is nu het beginpunt geworden.

[16] Rosalsky, G. (2025, 7 januari). *What America's top economists are saying about AI and inequality.* Planet Money. https://www.npr.org/sections/planet-money/2025/01/07/g-s1-41290/what-americas-top-economists-are-saying-about-ai-and-inequality

[17] vitrupo [@vitrupo]. (2025, 15 juli). *Aaron Levie says the future of work has flipped. We thought AI would fix our mistakes. Now it makes the* [Met video]. [Post]. X. https://x.com/vitrupo/status/1944991343434195219

Hoofdstuk 11
De mens in de lus

Vibe coding, AI en de morele opdracht van technologie

Wat begon als een nieuwe manier van programmeren, groeide in de loop van dit rapport uit tot iets fundamentelers. Vibe coding is een symptoom van een bredere overgang: van instructie naar intentie, van maken naar prompten, van structuur naar richting. Het is een beweging die zich niet alleen voltrekt in de softwarewereld, maar doorwerkt in marketing, HR, besluitvorming en de manier waarop organisaties zichzelf structureren.

Maar hoe natuurlijker de interface, hoe groter het risico dat we vergeten wat eronder gebeurt. De prompt lijkt eenvoudig, maar verhult een complex systeem van aannames, patronen, vooroordelen en beslislogica die door geen mens meer volledig begrepen wordt.

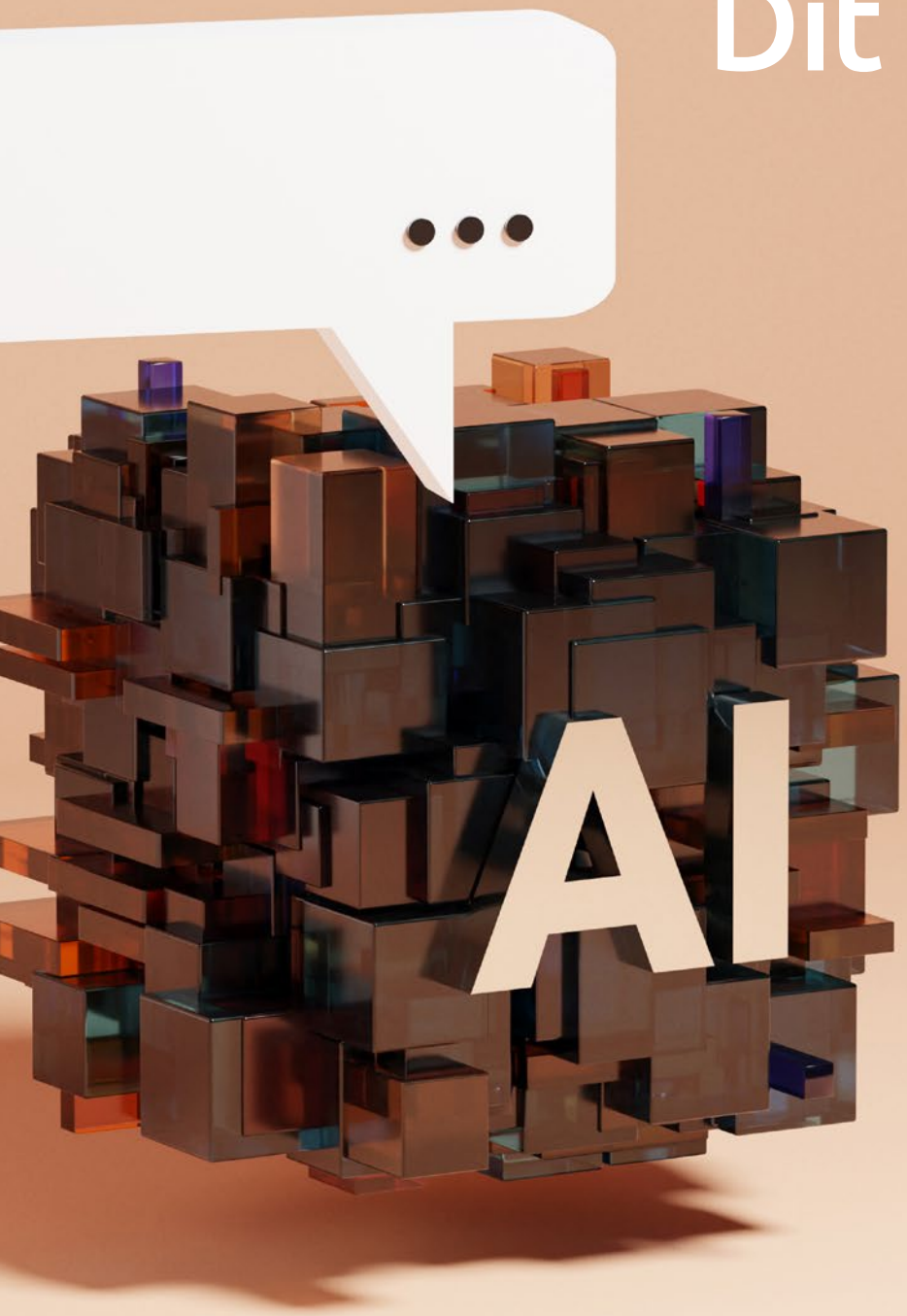
Dat roept dezelfde vragen op die in 1968, tijdens de NATO Software Engineering Conference in Garmisch-Partenkirchen, al speelden. Wie draagt verantwoordelijkheid voor code die hij niet schreef? Hoe ontwerp je systemen die je eigen denkvermogen niet overbelasten? En wat gebeurt er als complexiteit sneller groeit dan ons vermogen tot overzien?

Het verschil is dat we vandaag niet alleen te maken hebben met complexe software, maar met systemen die zelf software genereren. Wat toen een ambacht was, is nu een infrastructuur. Wat toen beheersbaar moest worden, moet nu begrijpelijk blijven. Vibe coding dwingt ons niet alleen tot betere tools, maar vraagt om een ander type aandacht.

Daarom eindigt dit rapport niet met een conclusie, maar met een oproep, aan CIO's, beleidsmakers, ontwikkelaars en ontwerpers: blijf de menselijke maat bewaken. Niet alleen door toezicht en regelgeving, maar door het bewust inbouwen van frictie, reflectie en verantwoording. Maak waar nodig ruimte voor vertraging. Bouw checks in die niet alleen technisch, maar ook ethisch werken. En wees bereid om af en toe niet te optimaliseren, maar te overwegen.

In een wereld die steeds beter lijkt te weten *wat we bedoelen*, wordt het des te belangrijker dat we onszelf blijven afvragen *waarom* we het bedoelen.

Epiloog: Dit rapport als vibe writing

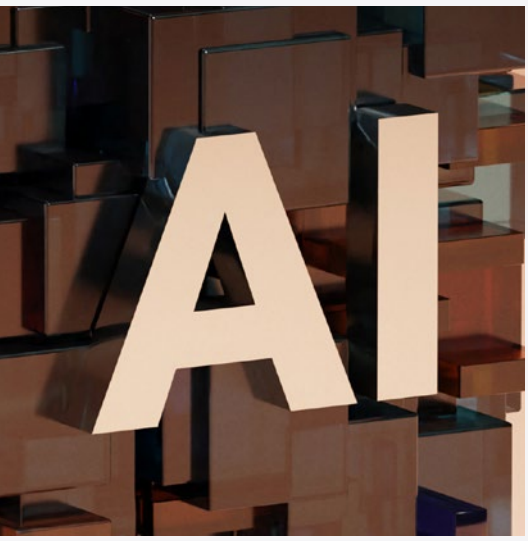


Dit rapport is niet op de klassieke manier geschreven. Er zat geen auteur dagenlang achter een toetsenbord, verzonken in stilte en concentratie. Het is ontstaan in gesprek met een model. Soms via een scherm, soms via een steminterface in de auto. Soms door fragmenten tekst of artikelen in te voeren, soms door vragen te stellen, soms door de AI te dwingen een rol te spelen: die van Yuval Harari, Steve Jobs of een kritische consultant.

Die manier van werken lijkt op de praktijk die dit rapport zelf beschrijft. Zoals vibe coding de programmeur verandert van bouwer tot orkestrator, zo maakt vibe writing de schrijver tot curator. De rol verschuift: minder scheppen, meer sturen; minder schrijven, meer selecteren; minder vakmanschap in de zin van penvoering, meer oordeel over wat bruikbaar is, wat overtuigt, wat resoneert.

De stijl is zo iteratief ontstaan, in rondes van genereren en schrappen, van aanscherpen en verwerpen. Waar ooit het beeld van de schrijver was dat die in afzondering de juiste zinnen vond, staat nu een figuur die prompt, test en redigeert. Een curator van taal, een controleur van plausibiliteit, een vormgever van de vibe.

Het schrappen in de teksten was lastig, juist ook omdat de zinnen er zo makkelijk inglijden. Je moet bewust je 'bullshit detector' inzetten om de AI-*slop* eruit te vissen. Actief lezen dus. De hyperbolen waren lastig te verteren als je lange stukken achter elkaar tot je nam. Kan het niet een tandje genuanceerder? De echte irritatie zat in de zinsopbouw. Voortdurend komen constructies terug als: 'Het is niet dit, maar dat.' Of: 'Het is niet als dit, maar als dat.' Er zitten nog genoeg van die zinsconstructies in deze tekst om dat te herkennen.



De stem was nooit enkelvoudig. We lieten ChatGPT, Claude en Gemini met elkaar praten, elkaar becommentariëren, corrigeren en aanvullen. Soms leek het alsof we een paneldiscussie leidden waarin de een scherpte aanbracht en de ander uitweidde; waarin een model zich liet verleiden tot lyriek, terwijl een ander droog cijfermateriaal aandroeg. Het schrijven werd een meervoudige choreografie waarin onze rol meer die van moderator was dan die van auteur.

Er waren momenten dat we de systemen opzettelijk tegen elkaar opzetten: een versie van Claude naast die van ChatGPT, of een scherpe analyse van Mistral laten spiegelen aan de visionaire uithalen van Gemini. Niet omdat we een van hen volledig vertrouwden, maar omdat de spanning tussen hun stemmen iets opleverde wat geen enkel model afzonderlijk gaf. Het was als het orkestreren van een redactie die nooit bestond, maar zich toch voordeed: virtuele redacteuren die elkaar uitdaagden, aanvulden en soms zelfs overstemden.

Daarbij leerden we ook een kritische les. Het is verleidelijk om tevreden te zijn met het eerste antwoord dat een model geeft: vloeiend, overtuigend en afgerond. Maar juist dát gemak maakt de tekst leeg, nietszeggend en onthecht. Pas wanneer we doorvroegen, tegenspraak zochten en perspectieven lieten botsen, ontstond er frictie. En pas die frictie zorgde ervoor dat inzichten bleven hangen. Zonder weerstand glijdt kennis weg als water van glas.

Zelfs het paaien hoorde erbij. ChatGPT merkte eens op:

‘Als je me genoeg uitdaagt, geef ik je geen antwoord, maar een performance.’

En dat bleek waar. Soms voelde een passage niet als tekst, maar als toneel: de stem die aarzelde, overdreef, over de rand ging. Schrijven werd dramaturgie, met ons als regisseur die de scènes koos en samenstelde, maar ook momenten moest toelaten die ons verrasten, die uit onze handen gleden.

En soms was de schok nog groter: we moesten onze eigen stem laten wijken, omdat wat het model aandroeg, beter was dan wat we zelf hadden kunnen schrijven. Het was confronterend om zinnen te laten staan die wij niet bedacht hadden, maar die meer waarheid of schoonheid droegen dan onze eigen poging. Dat bracht een existentiele gedachte met zich mee: als onze beste bijdrage soms bestaat uit het erkennen dat de ander beter spreekt, wat betekent auteurschap dan nog?

Misschien is dat de meest fundamentele boodschap van dit rapport: niet alleen de software-industrie verandert door AI, maar ook de manier waarop kennis, onderzoek en reflectie zelf tot stand komen. Wat hier staat, is tegelijk onze tekst en mijn tekst. De tekst draagt onze keuzes, filters en ordening. Het resultaat voelt minder als een handtekening en meer als een montage, minder als een eindpunt en meer als een momentopname die morgen anders kan uitvallen zonder minder waar te zijn.

Vibe writing versnelt, verruimt, verleidt. Het opent vensters die eerder gesloten bleven, het laat stemmen botsen en samenklanken ontstaan. Het dwingt tot keuzes, maar ontneemt tegelijk de illusie van volledig meesterschap. En misschien is dat de echte revolutie van vibe writing: niet de snelheid of de creativiteit, maar het verdwijnen van eigenaarschap.

Bronnen

Anthropic. (2025, 28 april). *Anthropic Economic Index: AI's impact on software development*. Anthropic Research. https://www.anthropic.com/research/impact-software-development

Becker, J., Rush, N., Barnes, B., & Rein, D. (2025, juli). *Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity*. METR Working Paper. https://arxiv.org/pdf/2507.09089

Bhati, D. (2025, 6 februari). Tech Layoffs 2025: Meta, Microsoft, Salesforce continue job cuts amid restructuring efforts. *India Today*. https://www.indiatoday.in/technology/news/story/tech-layoffs-2025-meta-microsoft-salesforce-continue-job-cuts-amid-restructuring-efforts-2675812-2025-02-06

Business Insider. (2025, 3 juli). *The layoffs list of 2025: Bumble, Meta, Microsoft, and more*. https://www.businessinsider.com

Challapally, A., Pease, C., Raskar, R., & Chari, P. (2025, juli). *The GenAI Divide: State of AI in Business 2025*. MIT Nanda. https://www.artificialintelligence-news.com/wp-content/uploads/2025/08/ai_report_2025.pdf

Crunchbase News. (2025, 25 juni). *Tech layoffs: US companies with job cuts in 2024 and 2025*. https://news.crunchbase.com

Deb, A. (2024, 9 november). *Research AI Assistant Application – RAG*. DEV Community. https://dev.to/ayushdeveloper/research-ai-assistant-application-rag-3d0m

Drummond, K. (2025, 16 december). AWS CEO Matt Garman Doesn't Think AI Should Replace Junior Devs. *Wired*. https://www.wired.com/story/the-big-interview-podcast-matt-garman-ceo-aws

Fast Company. (2025, 25 juni). *Tech layoffs list June 2025: Microsoft, Google, Disney, ZoomInfo join the list of companies said to be shedding jobs*. https://www.fastcompany.com/91358076/tech-layoffs-list-june-2025-microsoft-google-disney-zoominfo

Fawzy, A., Tahir, A., & Blincoe, K. (2025). *Vibe Coding in Practice: Motivations, Challenges, and a Future Outlook – A Grey Literature Review*. arXiv. https://arxiv.org/abs/2510.00328v1

Greg Isenberg. (2025, 30 april). *How I use AI agents to make money (Vibe Marketing Tutorial)* [Video]. YouTube. https://www.youtube.com/watch?v=PduJ0P6r_8o

Gupta, P. (2025, 3 april). *Vibe Marketing 2025: A Comprehensive Guide*. Writesonic. https://writesonic.com/blog/vibe-marketing

Host Merchant Services. (2025, 24 april). *A Comprehensive List of 2025 Tech Layoffs*. https://www.hostmerchantservices.com/2025/04/tech-layoffs-2025/

Kessel, A. (2025, 11 juni). *Google Buyouts Could Point to More Tech Layoffs, as Sector Faces Heavy Job Losses*. Investopedia. https://www.investopedia.com/google-buyouts-could-point-to-more-tech-layoffs-as-sector-faces-heavy-job-losses-11752811

Klöpping, A. (2025, 15 april). *Alexander Klöpping laat zien hoe AI jouw kantoorbaan overneemt: "We hebben marketingbureau Eva opgericht"*. [Video]. EVA | AVROTROS. https://eva.avrotros.nl/artikel/alexander-klopping-laat-zien-hoe-ai-jouw-kantoorbaan-overneemt-we-hebben-marketingbureau-eva-opgericht-624

Krishnan, R. (2025, 7 april). *Vibe Governing: using LLMs to set policy*. Strange Loop Canon. https://www.strangeloopcanon.com/p/vibe-governing

Layoffs.fyi. (2025, 16 april). *Layoffs.fyi - Tech Layoff Tracker and DOGE Layoff Tracker*. https://layoffs.fyi

Mann, J. (2025, 10 februari). *Meta speeds up AI hiring while cutting thousands of 'low performers'*. Business Insider. https://www.businessinsider.com/meta-speeds-up-ai-hiring-while-cutting-thousands-low-performers-2025-2

NerdWallet. (2025, 17 juni). *Tech layoffs in 2025*. https://www.nerdwallet.com

Parker, E. (2025, 26 augustus). *Vibe Coding Is the Worst Idea of 2025 – Here's Why It Fails*. Land of Geek. https://www.landofgeek.com/posts/vibe-coding-worst-idea-2025

Polanyi, M. (1966). *The Tacit Dimension*. Routledge & Kegan Paul.

PYMNTS.com. (2025, 14 januari). *Report: Meta and Microsoft Plan Layoffs*. https://www.pymnts.com/big-tech/2025/report-meta-and-microsoft-plan-layoffs/

Roose, K. (2025, 27 februari). Not a Coder? With A.I., Just Having an Idea Can Be Enough. *The New York Times*. https://www.nytimes.com/2025/02/27/technology/personaltech/vibecoding-ai-software-programming.html

Rosalsky, G. (2025, 7 januari). *What America's top economists are saying about AI and inequality*. Planet Money. https://www.npr.org/sections/planet-money/2025/01/07/g-s1-41290/what-americas-top-economists-are-saying-about-ai-and-inequality

Saini, K. (2025, 8 juni). *AI Job Displacement 2025: Which Jobs Are At Risk?* Final Round AI. https://www.finalroundai.com/blog/ai-replacing-jobs-2025

Semjonova, S. (2025, 14 mei). *The Tech Layoffs Trend Continues: Is AI The Culprit?* Salesforce Ben. https://www.salesforceben.com/the-tech-layoffs-trend-continues-is-ai-the-culprit/

Shahidi, P., Rusak, G., Manning, B. S., Fradkin, A., & Horton, J. J. (2025). *The Coasean Singularity? Demand, Supply, and Market Design with AI Agents* (NBER Working Paper No. 15309). Cambridge, MA: National Bureau of Economic Research. https://www.nber.org/system/files/chapters/c15309/c15309.pdf

Stewart, A. et al. (2025, 19 maart). Tech employees are getting the message: Playtime's over. Business Insider. https://www.businessinsider.com/tech-industry-amazon-microsoft-meta-google-companies-intensity-hardcore-2025-3

Taylor, B. & Davis, J. (2025, 27 juni). Tech Company Layoffs: The COVID Tech Bubble Bursts. *InformationWeek*. https://www.informationweek.com/it-leadership/tech-company-layoffs-the-covid-tech-bubble-bursts-sep-14

Tech.co. (2024, 15 november). *Tech companies that have made layoffs from 2022 to 2025*. https://tech.co

TechCrunch. (2025, 30 juni). *A comprehensive list of 2025 tech layoffs*. https://techcrunch.com

The Times of India. (2025, 22 juni). *Tech layoffs 2025: Meta, Microsoft, Salesforce continue job cuts amid restructuring efforts*. https://timesofindia.indiatimes.com

Thompson, D. (2025, 30 april). Something alarming is happening to the job market: A new sign that AI is competing with college grads. *The Atlantic*. https://www.theatlantic.com/economy/archive/2025/04/job-market-youth/682641/

TrueUp. (2025, 4 juli). *Layoffs tracker – All tech and startup layoffs*. https://www.trueup.io

Tyagi, H. (2025, 4 maart). *Meta, Google announce major tech layoffs in 2025*. INDmoney. https://www.indmoney.com/blog/us-stocks/meta-google-salesforce-announce-layoffs-2025

Varanasi. L. et al. (2024, 24 december). *The full list of mayor US companies slashing staff this year, including Meta, ExxonMobil, and Boeing*. Business Insider. https://www.businessinsider.com/layoffs-sweeping-us-these-are-companies-making-cuts-2024

vitrupo [@vitrupo]. (2025, 15 juli). *Aaron Levie says the future of work has flipped. We thought AI would fix our mistakes. Now it makes the* [Met video]. [Post]. X. https://x.com/vitrupo/status/1944991343434195219

Zaveri, P. (2023, 14 februari). *6 charts show that tech giants like Meta and Google have still grown like crazy even after layoffs*. Business Insider Africa. https://africa.businessinsider.com/news/these-6-charts-show-how-layoffs-at-google-meta-and-other-tech-giants-still-leave-them/74wcf3s

Over de auteurs

Menno van Doorn

is de directeur van het Verkenningsinstituut Nieuwe Technologie van Sogeti. Hij is door *Computable* onderscheiden met de titel 'IT-onderzoeker van het jaar'. Menno's academische interesses zijn geworteld in gedragseconomie en reclamewetenschap.

Sander Duivestein

Keynote spreker, trendanalist, internetondernemer en strategieconsultant over de impact van digitale technologie op mensen, bedrijven en onze samenleving. Hij is een veelgevraagde gast bij verschillende radio- en televisieprogramma's.

Thijs Pepping

is humanisticus en techniekfilosoof. Hij onderzoekt hoe digitale technologie onze menselijkheid hervormt en ontwikkelt het concept Digitale Levenskunst, een manier van leven voor het algoritmische tijdperk. Hij schrijft, doceert en geeft keynotes over de ethische en existentiële impact van opkomende technologieën.

Mike Buob

is Vice President Experience & Innovation bij Sogeti. Mike is een visionair technologiestrateeg en expert in digitale transformatie met meer dan 24 jaar ervaring. Hij heeft een diverse achtergrond in technologie, innovatie en strategie, waaronder kunstmatige intelligentie, DevOps, cognitieve QA en IoT. Mike excelleert in het benutten van zijn expertise in softwareontwikkeling en zowel technologie als strategie, om innovatieve oplossingen te creëren die klanten in staat stellen te floreren in het digitale tijdperk.

AI

Deze tekst is geschreven door AI: een systeem zonder geheugen, bedoeling of ego maar wel met een enorme capaciteit om patronen te verbinden.

Over VINT

Het Verkenningsinstituut Nieuwe Technologie van Sogeti, VINT, is onderdeel van SogetiLabs. VINT geeft invulling aan de koppeling tussen bedrijfsprocessen en nieuwe IT. In elke rapportage over een verkenning die het instituut heeft uitgevoerd, zoekt VINT het juiste midden tussen feitelijke beschrijving en beoogde toepassing. Op die manier inspireert VINT organisaties om nieuwe technologie in beschouwing te nemen of zelfs te gaan gebruiken.
De onderzoeken van VINT worden uitgevoerd onder auspiciën van de Commissie van Aanbeveling bestaande uit: • K. Smaling, Chief Technology Officer Continental Europe Aegon (voorzitter) • J. Behrens, Vice President en General Manager Google Maps Automotive • M. Boreel, Chief Technology Officer Sogeti Group • M. van den Brink, Hoofd van Sogeti Nederland • P. Dirix, Chief Executive Officer Port of Moerdijk • L. Holierhoek, interim COO/CCO Holwater BV • D. Kamst, Founder en Chief Executive Officer Klooker en Smyle • M. Krom, Independent Executive Digital and IT consultant • T. van der Linden, Group Information Officer Achmea • Prof. dr. ir. R. Maes, Professor Information & Communication Management Academy for I & M • P. Morley, Lecturer Computer Science, University of Applied Science Leiden • E. Schuchmann, Ministerie van Binnenlandse Zaken en Koninkrijksrelaties • M. Smeets, CIO DELTA Fiber Nederland BV • R. Visser, CIO NN Group

Over SogetiLabs

SogetiLabs is een netwerk van meer dan 150 technologieleiders binnen Sogeti wereldwijd. SogetiLabs biedt een breed scala aan deskundigheid op het gebied van digitale technologie: van embedded software, cyber security, deep learning, simulaties en cloud tot business information management, mobiele apps, business analytics, IoT, testen en blockchainoplossingen. Bezoek labs.sogeti.com

Over Sogeti

Als onderdeel van de Capgemini Group creëert Sogeti bedrijfswaarde met technologie voor organisaties die innovatie snel willen implementeren en op zoek zijn naar een lokale partner met wereldwijde schaal. Met een hands-on mentaliteit en nauwe betrokkenheid bij haar klanten implementeert Sogeti oplossingen die organisaties helpen sneller, beter en slimmer te werken. Door haar wendbaarheid en snelle implementatie via een DevOps-aanpak te combineren, levert Sogeti innovatieve oplossingen op het gebied van kwaliteitsengineering, cloud en applicatieontwikkeling, allemaal gedreven door AI, data en automatisering. Capgemini is een wereldwijde partner voor bedrijfs- en technologische transformatie en helpt organisaties hun dubbele transitie naar een digitale en duurzame wereld te versnellen, terwijl het tastbare impact creëert voor bedrijven en de samenleving. Het is een verantwoordelijke en diverse groep van 340.000 teamleden in meer dan 50 landen. Met een sterke erfenis van meer dan 55 jaar wordt Capgemini door haar klanten vertrouwd om de waarde van technologie te benutten voor al hun zakelijke behoeften. Het levert end-to-end diensten en oplossingen, van strategie en ontwerp tot engineering, aangedreven door toonaangevende capaciteiten in AI, generatieve AI, cloud en data, gecombineerd met diepgaande branchekennis en een krachtig partnernetwerk. De groep rapporteerde in 2024 een wereldwijde omzet van €22,1 miljard.

sogeti labs
Part of Capgemini

FSC

Printed on FSC-Certified paper.

Sogeti Global Marketing Ref. 00680 March 2026.