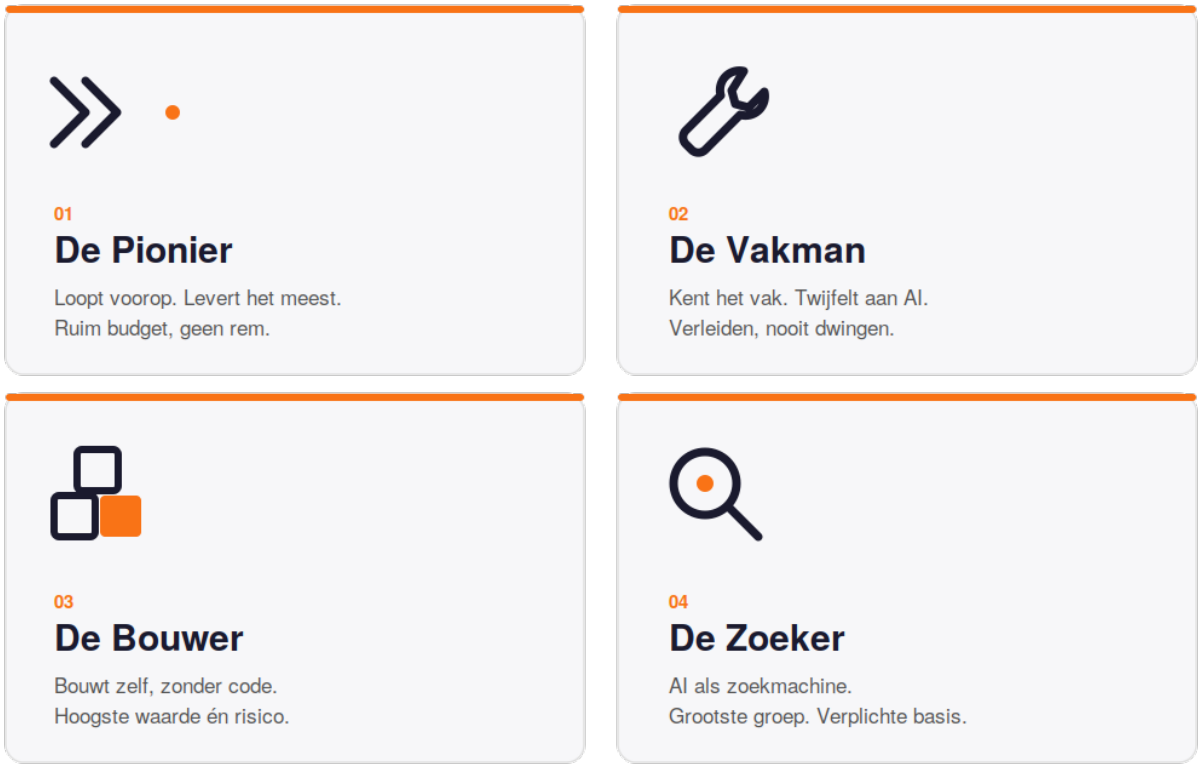
De vier AI-gebruikers in je organisatie

Waarom één AI-beleid voor iedereen drie van de vier mensen verkeerd behandelt



Marc Diks · juni 2026

marcdiks.nl

Executive summary

Er zijn vier typen AI-gebruikers in elke organisatie. Ze reageren tegengesteld op dezelfde knoppen. Wie iedereen hetzelfde behandelt, behandelt drie van de vier verkeerd.

Ik zie in de praktijk vier groepen ontstaan rond AI. De Pionier loopt voorop en levert de meeste output. De Vakman is de ervaren ontwikkelaar die AI nog op afstand houdt. De Bouwer bouwt zelf werkende tools zonder codeerachtergrond. En de Zoeker, veruit de grootste groep, gebruikt AI als een betere zoekmachine.

De fout die de meeste organisaties maken is dat ze deze vier behandelen als één probleem met één oplossing. Één beleid, één training, één set regels. Maar een tokenbudget dat de Pionier verstikt, is precies het focusfilter dat de Bouwer scherp houdt. Een verplichte training die de Zoeker eindelijk wakker schudt, bevestigt bij de Vakman het gevoel dat AI hem wordt opgedrongen. Dezelfde knop, tegengesteld effect.

Dit paper beschrijft de vier typen, hoe je elk afzonderlijk behandelt om er het maximale uit te halen, en het overkoepelende framework: drie knoppen die je per type bewust een andere kant op draait.

Vier mensen, vier werelden

Loop een willekeurige organisatie binnen en kijk hoe mensen AI gebruiken. Je ziet niet één gedrag, je ziet er vier. De ene developer draait Claude Code en Codex naast elkaar en verwerkt zijn backlog in recordtempo. Een collega vlak ernaast pakt AI er alleen bij voor een snippet of een bug, en sluit de tool daarna weer. Iemand op een businessafdeling bouwt zonder codeerkennis een tool die zijn hele team tijd bespaart. En de grootste groep typt af en toe een vraag in alsof het Google is.

Deze vier mensen vragen niet om hetzelfde. Ze hebben een tegengestelde aanpak nodig. Toch rollen organisaties standaard één AI-beleid uit over iedereen. Het gevolg: de voorhoede wordt afgeremd, de twijfelaar wordt verder de loopgraven in gejaagd, de bouwer creëert ongecontroleerd risico, en de grootste groep blijft aan de oppervlakte hangen.

In dit paper neem ik je mee langs de vier typen. Per type beschrijf ik wie het is, wat de waarde en het risico is, en vooral: hoe je ze behandelt om er het maximale uit te halen. Daarna breng ik het samen in één framework dat verklaart waarom één beleid per definitie niet kan werken.

Type 1 — De Pionier

De innovatieve voorhoede-developer

Wie is dit?

De Pionier is de ontwikkelaar die vooroploopt. Terwijl de rest nog leest over een nieuwe tool, draait die er al een week mee. Dit is degene die Claude Code en Codex naast elkaar gebruikt, die een nieuw model uitprobeert op de dag van release, en die zonder aarzelen drie aanpakken naast elkaar zet om te zien welke wint.

Het kenmerkende is niet hoeveel deze persoon praat met AI, maar hoe diep AI in het werk zit. Geen chatbot die er soms bij wordt gepakt, maar een vast onderdeel van het ontwikkelproces. Deze developer werkt vanuit een backlog en zet AI in om die sneller leeg te trekken. Het resultaat is zichtbaar: meer tickets, meer code, in dezelfde tijd.

Ja, deze groep verbruikt veel tokens. In absolute cijfers is dit de duurste groep van de organisatie. Maar dat verbruik is geen verspilling, het is leergeld en productiecapaciteit tegelijk. Wie het meest verbruikt, levert hier ook het meest op.

> **Het tokenverbruik van de Pionier is geen kostenpost maar output. De duurste groep is tegelijk de groep die het meest oplevert. Reken het door op resultaat, niet op verbruik.**

Wat is de waarde?

De ROI is hier het meest direct meetbaar van alle vier de typen: verwerkte tickets per sprint, opgeleverde code, en tokenverbruik als proxy voor activiteit. Deze groep verschuift de productiviteitsgrens van je hele ontwikkelafdeling. En ze doen meer dan code leveren: ze ontdekken wat werkt, zodat de organisatie niet bij elke nieuwe tool opnieuw het wiel hoeft uit te vinden.

Hoe behandel je ze?

- **Geef een ruim plafond, geen rem.** Stel het tokenbudget zo hoog in dat het zelden geraakt wordt. Elke minuut die naar het aanvragen van budget gaat, gaat niet naar output. Het budget is hier een veiligheidsnet, geen drempel.
- **Zie tokenverbruik als investering, niet als kostenpost.** De rekening is het hoogst, maar de opbrengst ook. Reken het door op output. Twee keer zoveel verbruik met drie keer zoveel tickets is geen kostenprobleem maar rendement.

- **Geef ze een mentorrol.** Ze leveren niet alleen output, ze trekken anderen mee. Maak het opleiden van collega's expliciet onderdeel van hun rol. Dit is je belangrijkste hefboom om de Vakman in beweging te krijgen.
- **Faciliteer, bemoei je verder niet.** Deze groep weet wat ze doet. Het grootste risico is dat je ze in de weg gaat zitten met onnodige goedkeuringen. Geef toegang, budget en vertrouwen, en haal obstakels weg.

Het doel is geen kleine elite, maar groei: zoveel mogelijk developers naar dit niveau tillen. De Pionier van vandaag is de mentor voor de Pionier van morgen.

Type 2 — De Vakman

De ervaren developer die AI op afstand houdt

Wie is dit?

Dit is de ontwikkelaar die AI er wel bij pakt, maar alleen voor losse klussen. Een snippet, een stuk uitleg, een bug. Daarna gaat de tool weer dicht. Vaak gaat het om de wat oudere, ervaren ontwikkelaars. Ze hebben hun vak geleerd zonder AI en zijn er goed in. Juist daar zit de wrijving: ze vertrouwen AI nog niet en zien het eerder als bedreiging dan als hulp.

Dit is geen onwetendheid en geen luiheid. Het is wantrouwen, gevoed door het gevoel dat er iets op het spel staat.

Waarom deze groep zo waardevol is

Onderschat dit type niet. Deze mensen weten nog hoe je echt bouwt. Ze kennen de fundamenten, herkennen een slechte architectuurkeuze, en kunnen de output van een AI nog goed valideren. Dat is precies de vaardigheid die in de andere groepen het snelst verdwijnt.

> **Zet je de Vakman om naar Pionier, dan krijg je de beste AI-ontwikkelaar die er is: de snelheid van AI gecombineerd met het oordeel van een doorgewinterde developer.**

Wat is het risico als ze blijven staan?

Twee problemen versterken elkaar. De productiviteitskloof met de Pioniers groeit elke maand, tot ze simpelweg worden ingehaald. En je betaalt dubbel: de organisatie schaft AI-licenties aan die deze groep nauwelijks benut, terwijl het werk trager gaat dan het zou kunnen. Stilstand is hier niet gratis.

Hoe behandel je ze?

- **Nooit dwingen.** Dit is de belangrijkste regel voor dit type. Verplichte trajecten en druk van bovenaf werken averechts. Dwang bevestigt precies het gevoel dat AI een bedreiging is die hen wordt opgedrongen. Hier verleid je, je dwingt niet.
- **Laat collega's het bewijs leveren.** Het overtuigendste argument komt niet van het management, maar van een vertrouwde collega die laat zien dat het werkt. Peer-bewijs verslaat overtuiging.
- **Bouw vertrouwen via kleine, gezamenlijke wins.** Zet de Vakman samen met een Pionier aan echt werk uit de eigen backlog. Elke win die ze zelf meemaken, knabbelt aan het wantrouwen. Het gaat om eigen ervaring, niet om beloftes.

- **Adresseer de bedreiging direct.** Omdat de kern angst is, benoem die expliciet. Maak duidelijk dat het doel is hun werk te verrijken en hun impact te vergroten, niet hen te vervangen.

Het doel is doorgroei naar Pionier. Dit type is de belangrijkste vijver waaruit nieuwe voorhoede-developers komen: de technische basis is er al, alleen het vertrouwen ontbreekt.

Type 3 — De Bouwer

De bouwende businessgebruiker

Wie is dit?

Dit is de zakelijke gebruiker die zelf werkende tools en apps bouwt, zonder formele codeerachtergrond. De Bouwer wacht niet op de IT-backlog, maar lost zijn eigen probleem op. Wie het probleem het beste kent, bouwt hier de oplossing, met prompting, agents, en no-code en low-code naast elkaar.

Het krachtige is de combinatie van domeinkennis en daadkracht. Maar precies dat wat dit type zo waardevol maakt, maakt het ook het meest riskant. Het is het enige type waar de hoogste waarde en het hoogste risico in dezelfde persoon samenkomen.

> **Bij de Bouwer zitten de hoogste waarde en het hoogste risico in dezelfde persoon. Daarom is governance hier geen rem, maar de voorwaarde om die waarde veilig te kunnen oogsten.**

Wat is de waarde?

De waarde komt uit vier hoeken tegelijk: snelheid zonder wachttijd op de IT-backlog, ontlasting van schaarse developers, innovatie dicht bij de business, en bewijslast in de vorm van werkende prototypes die een veel beter gesprek opleveren dan een PowerPoint vol aannames.

Wat is het risico?

Hier spelen alle vier de grote risico's tegelijk. **Shadow IT**: tools buiten het zicht van IT en governance. **Security en datagevaar**: geen besef van waar data heen gaat of welke poort wordt opengezet. **Onderhoudbaarheid**: valt de bouwer weg, dan kan niemand het overnemen. **Schaalbaarheid**: een prototype dat ongecontroleerd de productieomgeving in wordt geduwd. Bij dit type is governance geen rem maar een voorwaarde om de waarde veilig te kunnen oogsten.

Hoe behandel je ze?

- **Zet kaders, laat ze daarbinnen vrij bouwen.** IT bepaalt de grenzen rond data, tools en security. Binnen die kaders krijgt de Bouwer alle ruimte. Je beschermt de organisatie zonder de snelheid weg te halen.
- **Laat developers reviewen vóór productie.** Een prototype mag de Bouwer zelf maken, maar de stap naar productie loopt langs een echte developer die op security, onderhoudbaarheid en schaalbaarheid controleert.

- **Laat IT overnemen bij opschaling.** Slaat een tool aan, draag het dan over. De Bouwer bewijst het probleem, IT maakt het robuust.
- **Gebruik budget als focusfilter.** Tegenovergesteld aan de Pionier krijgt de Bouwer een strak budget. Wie meer wil, legt aan zijn leidinggevende uit wát hij wil bouwen. Die drempel filtert serieuze projecten van hobbyprojecten die nooit live gaan.

De onderscheidende vaardigheid

Wat maakt het verschil tussen een goede en een slechte Bouwer? Niet de technische kennis. Het zwaartepunt ligt vooraan: een scherp probleem kunnen definiëren, dan goede prompts schrijven, dan weten wanneer je IT moet inschakelen, en pas daarna het beoordelen van de output. Goed bouwen begint bij helder denken, niet bij code lezen. De schaarse vaardigheid is hier zakelijk inzicht, en dat is precies wat dit type van nature al heeft.

Type 4 — De Zoeker

ChatGPT als zoekmachine

Wie is dit?

Dit is veruit de grootste groep, de meerderheid. Voor de Zoeker is AI vooral een betere zoekmachine: een vraag erin, een antwoord eruit. Daarnaast wat licht tekstwerk, zoals een e-mail laten samenvatten of opstellen. Geen vervolgvragen, geen context, geen tweede iteratie.

Het belangrijkste om te begrijpen: ze blijven niet aan de oppervlakte uit angst, maar uit onbekendheid. Ze weten simpelweg niet wat er méér mogelijk is, hebben geen prikkel om verder te leren, en hun werk vraagt er op het eerste gezicht niet om. Het is geen weerstand, het is een blinde vlek.

> **Vakman en Zoeker lijken op elkaar, maar blokkeren om tegengestelde redenen. Bij de Vakman is het emotioneel: wantrouwen. Bij de Zoeker praktisch: onbekendheid. Daarom werkt dezelfde interventie bij de één averechts en bij de ander niet.**

Wat is de waarde en het risico?

De waarde per persoon is hier het laagst. Maar omdat dit veruit de grootste groep is, zit hier collectief de grootste onbenutte potentie van de hele organisatie. Het risico is subtiel: deze mensen denken dat ze al AI-gebruiker zijn. Je kunt geen honger creëren bij iemand die denkt dat hij al gegeten heeft.

Hoe behandel je ze?

Hier draait de aanpak honderdtachtig graden om ten opzichte van de Vakman. Waar je de ervaren developer nooit mag dwingen, mag en moet je deze groep juist wel een verplichtende basis geven. Zonder die duw komt de combinatie van onbekendheid en gebrek aan prikkel nooit los.

- **Begin met verplichte basistraining voor iedereen.** Omdat de blokkade onbekendheid is, geef je iedereen hetzelfde startpunt. Verplichting werkt hier wél, want je vult een lege plek, je duwt niet tegen een muur.
- **Laat collega's de inspiratie leveren.** Na de basis komt motivatie van laagdrempelige voorbeelden. “Kijk wat ik hiermee in tien minuten deed” doet meer dan welke module ook.

- **Reik kant-en-klare prompts en use-cases per functie aan.** Niet “ga zelf ontdekken”, maar “hier zijn vijf dingen die voor jouw werk meteen waarde opleveren”.
- **Bouw AI in in de tools die ze toch al gebruiken.** Zit de functie in hun bestaande omgeving, dan wordt gebruik vanzelfsprekend in plaats van een extra handeling.

Voor deze groep werkt geen sprong, maar een trap: eerst de basis veilig op orde, dan een paar krachtige use-cases per functie, dan van incidenteel naar gewoonte. Een bescheiden stap per persoon is bij deze aantallen de grootste collectieve sprong die de organisatie kan maken.

Het framework

Waarom één AI-beleid voor iedereen niet werkt

De meeste organisaties behandelen AI-adoptie als één probleem met één oplossing. Één beleid, één training, één set regels, uitgerold over iedereen. Dat is precies waarom het misgaat. De vier typen reageren tegengesteld op dezelfde knoppen. Wat de Pionier vooruithelpt, remt de Bouwer af. Wat de Zoeker in beweging krijgt, jaagt de Vakman verder de loopgraven in.

> **Wie iedereen hetzelfde behandelt, behandelt drie van de vier verkeerd.**

De drie knoppen die je per type omdraait

Het framework draait om drie variabelen. Geen instellingen die je één keer goed zet, maar knoppen die je bewust per type een andere kant op draait. De oranje vlakken hieronder laten zien waar je streng of sturend bent: ze zitten verspreid, niet in één kolom.

	Pionier	Vakman	Bouwer	Zoeker
Budget	Ruim plafond	Volgt vanzelf	**Strak + verantwoording**	Niet relevant
Dwang	Niet dwingen, faciliteren	Nooit dwingen, verleiden	**Kaders verplicht, bouwen vrij**	**Verplichte basis mag en moet**
Governance	Minimaal	Minimaal	**Maximaal**	Basis-veiligheid

De drie knoppen per gebruikerstype. Oranje = streng of sturend, licht = ruimte.

Het scherpste contrast zit tussen Pionier en Bouwer bij budget. Bij de één is een ruim budget een investering in output, bij de ander is een strak budget een investering in focus. Dezelfde knop, tegengestelde stand, beide om de goede reden. Bij dwang draaien Vakman en Zoeker om: bij de Zoeker werkt verplichting wél, omdat er geen muur is om tegenaan te duwen, alleen een lege plek om te vullen.

De groeibeweging

De vier typen zijn geen vakjes waarin mensen vastzitten. Het zijn posities op een beweging met een richting. De Zoeker beweegt via een trap omhoog, en de gemotiveerden onder hen kunnen doorgroeien richting Bouwer. De Vakman is de belangrijkste vijver voor nieuwe Pioniers: de technische basis is er al, alleen het vertrouwen ontbreekt. De Pionier is geen eindstation maar een groeigroep die je zo groot mogelijk wilt maken.

Het doel is niet om iedereen Pionier te maken. Het doel is iedereen één positie te laten opschuiven. Bij de grootste groep is dat collectief de grootste sprong; bij de Vakman is het kwalitatief de waardevolste.

De Pionier als spil

Eén element bindt het hele framework samen: de Pionier is niet alleen de meest productieve groep, maar ook de motor achter de beweging van de anderen. Voor de Vakman is de Pionier het levende bewijs en de hands-on mentor die het wantrouwen wegneemt. Voor de Bouwer is de Pionier de developer die prototypes reviewt en opschaalt. Voor de Zoeker is de Pionier de collega die laagdrempelige wins deelt na de basistraining.

Daarom is de mentorrol van de Pionier geen extraatje maar een kernfunctie. Wie de Pioniers alleen als individuele productiemachines inzet, laat hun grootste hefboom liggen: hun vermogen om de andere drie typen in beweging te brengen.

Conclusie en aanbevelingen

AI-adoptie is geen kwestie van één knop omzetten voor de hele organisatie. Het is een kwestie van per groep de juiste knop de juiste kant op draaien. Concreet betekent dat:

1. **Stop met één AI-beleid.** Ontwerp per type, want de knoppen staan tegengesteld.
2. **Differentieer budget bewust.** Ruim voor Pioniers, strak met verantwoording voor Bouwers. Niet als bezuiniging, maar als sturing.
3. **Dwing selectief.** Verplicht waar geen weerstand zit (Zoeker), verleid waar wel weerstand zit (Vakman).
4. **Zet governance waar het risico zit.** Zwaar op de Bouwer, licht op de rest.
5. **Maak de mentorrol expliciet.** De Pionier is je belangrijkste adoptie-instrument, niet alleen je snelste developer.
6. **Stuur op beweging, niet op bestemming.** Eén positie opschuiven per type is de winst, niet iedereen tot Pionier maken.

Over de auteur

Marc Diks schrijft over AI-strategie, governance en digitale transformatie voor Nederlandse organisaties. Hij bouwt zelf productieapplicaties met AI-tools en deelt zijn praktijkervaring op zijn blog, LinkedIn en Substack.

Website: marcdiks.nl | LinkedIn: linkedin.com/in/marcdiks | X: x.com/marcdiks