KOFETKE.

02 / 10

VODNIKI O AI

Razlika med pogovorom, delovnim tokom, botom in agentom

Katera vrsta sistema je primerna za to delo?



AVTOMATIZACIJA IN AGENTI

Delovni tok za malo podjetje s sprožilci, odločitvami, stanjem in odobritvami.

SLOVENSKA IZDAJA

september 2026
Praktična uporaba AI

Viri preverjeni 21. septembra 2026.

Uporabna avtomatizacija se začne z majhno obljubo: ko nastopi določen dogodek, opravi določen del dela in pusti sled. Prispe obrazec za račun, na projektni tabli se pojavi kartica ali stranka zaprosi za termin. Sistem kopira ustrezna polja, preveri pravilo in rezultat postavi tja, kjer ga lahko nekdo uporabi. To je že dragoceno. Odpravi prepisovanje in poskrbi, da rutinsko delo ni toliko odvisno od tega, ali si ena oseba zapomni naslednji klik.

Težava nastane, kadar tako preprosto ureditev opišemo kot »agenta« in ji damo širok dostop, preden je kdo določil, kaj šteje za uspeh, kaj za podvojitev ali kdo lahko odobri nepovraten korak. Oznaka ne izboljša zasnove. Delovni tok z jasnimi omejitvami je običajno uporabnejši od spretnega pomočnika z nejasnimi dovoljenji.

To poglavje loči štiri dele, ki jih pogosto združujemo: pogovorni vmesnik, delovni tok, bot in agent. Pojasnjuje tudi manj vidne mehanizme: sprožilce, API-je, spletne kljuke, stanje, ponovne poskuse in meje odobravanja. Ti deli odločajo, ali sistem tiho pomaga ali pa ustvari urejeno videti zmedo.

Dogodek in odziv

Sprožilec je dogodek, ki začne delo. Lahko je načrtovani čas, oddan obrazec, nova vrstica v podatkovni zbirki ali zahteva HTTP. API oziroma programski vmesnik je dokumentiran način, kako en program drugega zaprosi za branje ali spremembo podatkov. Spletna kljuka deluje v obratni smeri: aplikacija ob dogodku pošlje

zahtevo HTTP na podani naslov. Zapierjev Catch Hook na primer prejemnemu delovnemu toku dodeli enoličen URL, na katerega lahko sprožilna aplikacija pošlje zahtevo POST.[20]

Ta razlika ima praktične posledice. Načrtovano preverjanje vsakih nekaj minut sprašuje: »Ali se je kaj spremenilo?« Spletna kljuka ob dogodku sporoči: »To se je spremenilo.« Make spletne kljuke opisuje kot takojšnje sprožilce, v nasprotju z načrtovanimi sprožilci, ki storitev občasno sprašujejo po podatkih.[19] Make, Zapier in n8n lahko uredijo obe vrsti povezave. Izbira naj sledi izvornemu sistemu. Če koledar ponuja zanesljivo obvestilo o spremembi, z vsakominutnim preverjanjem tratimo klice in otežujemo obravnavo podvojitev. Če vir ne ponuja toka dogodkov, je previdno načrtovano preverjanje morda edina možnost.

Osnovni delovni tok je mogoče narisati brez AI:

```
nova oddaja obrazca
    -> preveri obvezna polja
    -> ustvari zapis naloge
    -> pošlji potrditev
    -> shrani vnos v revizijsko sled
```

Vsaka puščica mora prenašati znano obliko podatkov. Oddaja obrazca lahko vsebuje **name**, **email**, **service**, **message** in ID oddaje. Preverjanje odloči, ali so obvezna polja prisotna in ali vrednosti ustrezajo dovoljenemu zapisu. Vnos v revizijski sledi zabeleži, kaj se je zgodilo, kdaj, iz katerega vira in s katerim identifikatorjem. Tak zapis je uporabnejši od okrasne nadzorne plošče, ko lastnik vpraša: »Zakaj je bilo to sporočilo poslano dvakrat?«

Najzanesljivejši delovni tokovi so podobni temu diagramu. Začnejo se z ozkim vhodom, ga preoblikujejo z običajnimi pravili in ustvarijo viden rezultat. Platforma za avtomatizacijo je izbira izvedbe, ne sama zasnova. Skript za preglednico, scenarij Make, delovni tok Zapier ali tok n8n lahko vsi uresničijo isti dogovor.

Štiri sorodne, vendar različne stvari

Pogovor je mesto, kjer oseba piše in prejema odgovore. Lahko sploh ničesar ne kliče. Pogovor v Telegramu z jezikovnim modelom se lahko zdi kot pomočnik, vendar nima zmožnosti videti koledarja ali ustvariti naloge, dokler nekdo ne poveže orodij in ne dodeli dovoljenj.

Delovni tok je zaporedje z znanimi vejami. »Ko prispe odobren obrazec, ustvari osnutek naloge; če je polje proračuna prazno, ga usmeri v pregled« je delovni tok. Lahko ima veliko vej, a avtor jih lahko našteje. Njegove odločitve bi morale biti ponovljive iz shranjenih vhodov in pravil.

Bot je program, povezan z računom za sporočanje ali storitvenim računom. Prejema izbrane dogodke ter odgovarja ali kliče API-je. Bot lahko izvaja delovni tok, vendar beseda bot ne pove ničesar o tem, koliko lahko bere, si zapomni ali spremeni. Je vmesnik in izvajalna identiteta, ne varnostno zagotovilo.

Agent doda zanko, ki glede na cilj in informacije, ki jih vidi, izbira med orodji ali naslednjimi koraki. Lahko preišče zbirko znanja, razvrsti zahtevo, pripravi osnutek odgovora in zaprosi za človeško

odločitev. To je lahko koristno, kadar se pot zares spreminja. Hkrati razširi število možnih dejanj, zato je treba bolj premišljeno zasnovati dovoljenja in pogoje ustavitve.

Sestavni del	
Pogovor	
Najprimernejša uporaba	Vprašanja in osnutki
Slog odločanja	Pogovor
Uporabna meja	Ne more tiho ukrepati drugje
Delovni tok	
Najprimernejša uporaba	Ponavljajoča se rutina
Slog odločanja	Izrecna pravila
Uporabna meja	Vhodi in izhodi imajo shemo
Bot	
Najprimernejša uporaba	Sporočila ali dogodki storitev
Slog odločanja	Odziv na dogodek
Uporabna meja	Dovoljenja platforme omejujejo vidnost
Agent	
Najprimernejša uporaba	Spremenljivo raziskovanje ali triaža
Slog odločanja	Izbira orodij znotraj cilja
Uporabna meja	Seznam dovoljenih orodij, proračun in vrata za odobritev

Agent ne bi smel nadomestiti pravila, ki že obstaja. Pravilo lahko odloči, ali se skupni znesek računa ujema z vsoto njegovih postavk. Lahko zavrne e-poštni naslov brez znaka @. Zahtevo znane stranke lahko usmeri v imenovano čakalno vrsto. Jezikovni model lahko pomaga razložiti nejasno zahtevo v prostem besedilu, vendar mora koraku, ki temelji na pravilih, vrniti strukturirane predloge, namesto da bi si izmišljal knjigovodske odločitve ali odločitve o dovoljenju.

Stanje je spomin procesa

Stanje pomeni dejstva, ki jih mora sistem ohraniti med dogodki. Delovni tok, ki prebere en obrazec in takoj ustvari eno nalogo, potrebuje malo stanja. Delovni tok, ki se mora izogniti podvojitvam, počakati na odobritev, nadaljevati po izpadu storitve in poročati o dogajanju, ga potrebuje več.

Najmanjši zapis stanja bi lahko bil takšen:

```
source_event_id: form-2026-0912-184
received_at:     2026-09-12T14:07:22Z
status:          awaiting_review
attempt_count:   0
output_task_id:  null
approval_by:     null
```

ID izvornega dogodka je sidro. Preden ustvari novo nalogo, delovni tok poišče ta ID. Če je že ustvaril izhodno nalogo, se ustavi in vrne prejšnji rezultat. Ta lastnost se imenuje idempotentnost: ponovitev istega vhoda ne ustvari novega nenamernega učinka. Pomembna je, ker storitve za dogodke ponavljajo poskuse. Slack

od prejemnika dogodkov zahteva, da zahtevo potrdi z odgovorom HTTP 2xx v treh sekundah, in dokumentira ponovne poskuse po neuspehu.[15] Počasen ali sesut prejemnik lahko zato znova vidi isti dogodek.

Pravi prvi odziv na dogodek je pogosto namenoma dolgočasen: preverite podpis, kjer ga platforma ponuja, shranite ID dogodka in sklic na vsebino, dodajte opravilo v čakalno vrsto, nato potrdite prejem. Čakalna vrsta je zadrževalna vrsta za delo, ki lahko traja dlje, kot dovoljuje izvorna platforma. Izvajalni proces lahko nato obdeluje po eno opravilo, ponovi začasne neuspehe in zabeleži rezultat. Slackova navodila pravijo, naj obdelave in odziva ne izvajamo v istem procesu, ki prejme dogodek, ter po prejemu priporočajo čakalno vrsto.[15]

Ponovni poskusi potrebujejo politiko. Začasna omrežna prekinitev lahko upraviči tri razmaknjene poskuse. »Dostop zavrnjen« se sploh ne sme ponavljati; gre za težavo v konfiguraciji. Napačno zapisan datum naj gre v čakalno vrsto za pregled, ne pa večkrat v isti API. Vsako poskušeno dejanje potrebuje stabilen ključ idempotentnosti ali izvorni ID, časovni žig, izid in človeku razumljivo napako. Brez teh polj se sistem ponovnih poskusov spremeni v ustvarjalnik podvojitev.

Avtonomiji postavite ograjo

Agent je najvarnejši, kadar je njegovo predlagano dejanje ločeno od dejanja samega. Lahko sestavi osnutek, razvrsti zahtevo ali priporoči naslednji korak, nato pa se ustavi. Človeški lastnik pred odobritvijo zunanjega učinka vidi ustrezen vir, predlagane parametre in cilj.

n8n dokumentira vzorec človeškega pregleda, pri katerem se delovni tok ustavi, preden agent AI uporabi določeno orodje; pregledovalec lahko dejanje odobri ali zavrne.[13] Isto načelo deluje tudi zunaj n8n. Uporabna zahteva za odobritev vsebuje točno predlagano operacijo, ne nejasnega stavka, kot je »Pošlji obvestilo.« Vsebovati mora:

```
Predlagano dejanje: ustvari nalogo
Cilj:               Projektna tabla / septembrski sprejem
Naslov:             Odgovori Ani glede datotek z logotipom
Vir:                ID e-poštnega sporočila 18c7...
Rok:                rok ni predlagan
Razlog:             sporočilo prosi za popravljene datoteke
Odobri / zavrni / uredi
```

Odobritev je zlasti primerna za pošiljanje sporočil, spreminjanje zapisov, brisanje podatkov, nakupe, deljenje datotek zunaj organizacije ali spreminjanje dostopa. n8n med bolj tvegana orodja za selektivni pregled uvršča pošiljanje sporočil, spreminjanje zapisov in brisanje podatkov.[13] Enako previdnost je smiselno uporabiti tudi za delovni tok malega podjetja, čeprav predpis tega ne zahteva.

Tudi dovoljenja naj bodo ozka. Raziskovalni pomočnik lahko potrebuje bralni dostop do ene mape v skupni rabi in dovoljenje za ustvarjanje osnutkov v določenem seznamu nalog. Ne potrebuje pooblastila za spreminjanje bančnih podatkov, vabljenje uporabnikov v vsak delovni prostor ali pošiljanje sporočil z lastnikovo identiteto. Dostop je del sistemske specifikacije, poleg navodila modelu in izbire modela.

Izmišljen sistem sprejema, preizkušen pred zaupanjem

Predstavljajte si izmišljen oblikovalski studio z dvema osebama. Lastnik prejema povpraševanja o projektih prek obrazca, Telegrama in e-pošte. Studio želi vsakodnevni pregled novega dela, ne da bi izgubil izvirna sporočila ali po naključju obljubil datum dobave.

Prva različica ne vsebuje agenta. Vsak vir ustvari zapis v preglednici sprejema z vrsto vira, izvornim ID-jem, pošiljateljem, časom prejema, izvirno povezavo, kratkim izvlečkom in stanjem **new**. Deterministično pravilo zavrne oddaje brez kontaktnih podatkov. Drugo pravilo obstoječi izvorni ID označi kot podvojitev. Ob 16:30 delovni tok združi preostale zapise po stanju in lastniku pošlje povzetek s povezavami nazaj na njihov vir. Lastnik, ne sistem, odloči, katero povpraševanje postane naloga za stranko.

Druga različica doda omejen korak z jezikovnim modelom. Za vsako novo sporočilo vrne JSON s **topic**, **urgency**, **suggested_next_action** in **uncertainty**. Preverjevalnik sprejme samo znane teme in ob predlogu ohrani izvirno sporočilo. Če se model ne more odločiti, mora biti **uncertainty high**; element ostane v običajni čakalni vrsti za pregled. Sistem sme pripraviti osnutek naslova naloge, nikoli pa poslati ponudbe ali ustvariti rezervacije v koledarju.

Tretja različica po odobritvi dovoljuje eno dejanje: ustvari nalogo na projektni tabli studia. Kartica za odobritev prikaže točna polja in povezavo do vira. Sprejeta kartica shrani identiteto lastnika in

čas. Zavrnjena kartica shrani razlog, ki kasneje pomaga izboljšati pravilo ali navodilo. To je omejena avtonomija: sistem lahko razlaga vhod, vendar se njegovo pooblastilo konča pred zavezo.

Primer neuspeha, ki ga je vredno vaditi

Pogost neuspeh se začne s prekoračitvijo časovne omejitve pri spletni kljuki. Ponudnik obrazca pošlje dogodek. Prejemni delovni tok ustvari nalogo, vendar za odgovor potrebuje predolgo. Ponudnik poskus ponovi. Delovni tok ustvari drugo nalogo in lastnik vidi dve potrditvi. Nekdo eno ročno izbriše, zapisi pa ostanejo neusklajeni.

Popravek se začne z evidenco dogodkov. ID dogodka shranite pred vsakim zunanjim dejanjem. Ob vsakem zagonu atomarno rezervirajte ta ID s stanjem **processing**. Če prispe druga kopija, vrnite obstoječe stanje. Nastali ID naloge pripnite šele po uspešni ustvaritvi naloge. Če ustvarjanje po nejasnem omrežnem odgovoru ne uspe, pred ponovnim poskusom na cilju poiščite shranjeni zunanji ID. Nato isti testni dogodek dvakrat izvedite v testnem delovnem prostoru in preverite, da obstaja le ena naloga.

Prav tako so pomembna druga preverjanja: oddajte nepopoln obrazec, prekličite dovoljenje, simulirajte omejitev števila zahtev API-ja, zavrnite odobritev, naredite cilj nedosegljiv in vstavite sporočilo, ki pomočnika prosi, naj ne upošteva svojih pravil. Sistem, ki prestane le uspešno pot, ni bil ovrednoten.

Praktično merilo je sledljivost, ne vznemirjenje. Pri vzorcu resničnih ali skrbno izmišljenih dogodkov bi moral lastnik znati odgovoriti: Kateri dogodek je to začel? Katera polja so bila prebrana? Katero pravilo ali izhod modela je oblikoval predlog? Katero dejanje se je zgodilo? Kdo ga je odobril? Ali je mogoče isti dogodek ponoviti brez podvojitve? Če so odgovori vidni, avtomatizacija postane orodje, ki ga je mogoče popraviti. Če manjkajo, je sistem hiter le do prve napake.

Viri

Številke virov sledijo skupnemu seznamu. Povezave vodijo do izvirnih dokumentov. Pogoji uporabe in razpoložljivost storitev se lahko spremenijo.

[13] n8n: human review for tools
https://docs.n8n.io/build/integrate-ai/ai-examples/human-in-the-loop-for-tools

[15] Slack: Events API
https://api.slack.com/apis/connections/events-api

[19] Make: webhooks
https://help.make.com/webhooks

[20] Zapier: webhook triggers
https://help.zapier.com/hc/en-us/articles/8496288690317-Trigger-Zap-workflows-from-webhooks