PUBLIKACJE / ARTYKUŁ PRZEGLĄDOWY

ASYSTENCI AI W WYTWARZANIU OPROGRAMOWANIA

Przegląd badań nad produktywnością, jakością i bezpieczeństwem kodu

AUTOR

Tomasz Załucki

DATA PUBLIKACJI

16 listopada 2026 r.

CZAS CZYTANIA

12 min

STRESZCZENIE

Artykuł zestawia wyniki eksperymentów kontrolowanych i badań ankietowych z lat 2022-2025 dotyczących wpływu asystentów AI na pracę programistów. Pokazuje rozbieżność między badaniami laboratoryjnymi a badaniem na dojrzałych projektach, omawia wyniki na poziomie organizacji oraz ryzyka dla bezpieczeństwa kodu, a na koniec formułuje wnioski dla zespołów i programów szkoleniowych.

Słowa kluczowe: asystenci kodu, sztuczna inteligencja, produktywność programistów, bezpieczeństwo kodu, AI Act

1. WPROWADZENIE

Asystenci programowania oparci na dużych modelach językowych w ciągu trzech lat przeszli drogę od narzędzia eksperymentalnego do standardowego wyposażenia stanowiska pracy. Decyzje o ich wdrażaniu zapadają zwykle na podstawie deklaracji dostawców i wrażeń użytkowników. Tymczasem istnieje już kilka badań o kontrolowanym charakterze, których wyniki są ze sobą na pierwszy rzut oka sprzeczne: od przyspieszenia pracy o ponad połowę po jej spowolnienie o jedną piątą.

Celem artykułu jest zestawienie tych wyników i wskazanie, w jakich warunkach każdy z nich został uzyskany. Przegląd obejmuje trzy obszary: produktywność pojedynczego programisty, wyniki na poziomie organizacji oraz bezpieczeństwo generowanego kodu.

2. SKALA UPOWSZECHNIENIA

W ankiecie Stack Overflow z 2025 r. 84% respondentów zadeklarowało, że używa narzędzi AI w procesie wytwarzania oprogramowania lub planuje ich użycie, wobec 76% rok wcześniej[1]. Równocześnie więcej respondentów deklaruje brak zaufania do trafności odpowiedzi narzędzi AI (46%) niż zaufanie (33%), a tylko 3% określa swoje zaufanie jako wysokie[1]. Raport DORA z 2025 r., oparty na odpowiedziach blisko pięciu tysięcy specjalistów, podaje poziom adopcji AI wśród osób zawodowo wytwarzających oprogramowanie na 90%[2].

Zestawienie tych liczb pokazuje stan, który warto nazwać wprost: narzędzia są używane powszechnie przez osoby, które w dużej części nie ufają ich wynikom. Pytanie o rzeczywisty wpływ na pracę jest więc pytaniem empirycznym, a nie kwestią nastawienia.

3. PRODUKTYWNOŚĆ W EKSPERYMENTACH

3.1. Zadanie laboratoryjne

Peng i współautorzy przeprowadzili eksperyment kontrolowany, w którym rekrutowanym programistom polecono jak najszybciej zaimplementować serwer HTTP w języku JavaScript. Grupa korzystająca z asystenta GitHub Copilot ukończyła zadanie o 55,8% szybciej niż grupa kontrolna[3]. Jest to wynik najczęściej przywoływany w materiałach promocyjnych. Jego ograniczeniem jest charakter zadania: krótkie, dobrze zdefiniowane, realizowane od zera, bez istniejącej bazy kodu i bez wymagań jakościowych wykraczających poza przejście testów.

3.2. Eksperymenty terenowe w firmach

Cui i współautorzy przeanalizowali trzy randomizowane eksperymenty terenowe przeprowadzone w Microsoft, Accenture i anonimowej firmie z listy Fortune 100, obejmujące łącznie 4867 programistów. Dostęp do asystenta kodu wiązał się ze wzrostem liczby ukończonych zadań o 26,08%, przy czym większe korzyści odnotowano u osób mniej doświadczonych[4]. Błąd standardowy tego oszacowania wynosi 10,3%, co oznacza znaczną niepewność. Miarą wyniku była liczba ukończonych zadań, a nie ich jakość ani późniejsze koszty utrzymania.

3.3. Doświadczeni programiści we własnych projektach

Odmienny wynik przyniosło badanie organizacji METR z 2025 r. Szesnastu programistów projektów otwartoźródłowych wykonało 246 zadań w dojrzałych repozytoriach, w których pracowali średnio od pięciu lat. Każde zadanie losowo przydzielano do wariantu z dozwolonym lub niedozwolonym użyciem AI. Dopuszczenie narzędzi AI wydłużyło czas wykonania zadań o 19%[5].

Równie istotny jest drugi wynik tego badania. Przed rozpoczęciem uczestnicy przewidywali, że AI skróci czas wykonania zadań o 24%. Po zakończeniu, mimo rzeczywistego spowolnienia, nadal szacowali, że narzędzia skróciły ten czas o 20% [5]. Subiektywna ocena przyspieszenia okazała się więc nie tylko niedokładna, ale skierowana w przeciwną stronę niż pomiar.

3.4. Jak pogodzić te wyniki

Tabela 1. Porównanie badań produktywności

BADANIE	UCZESTNICY I KONTEKST	WYNIK
Peng i in. (2023)	Rekrutowani programiści, jedno nowe zadanie, brak istniejącego kodu	Zadanie ukończone o 55,8% szybciej
Cui i in. (2024)	4867 programistów, praca bieżąca w trzech firmach	O 26,08% więcej ukończonych zadań
METR (2025)	16 doświadczonych programistów, 246 zadań w dojrzałych repozytoriach	Czas wykonania dłuższy o 19%

Źródło: opracowanie własne na podstawie przypisów 3, 4 i 5.

Badania te nie przeczą sobie, tylko mierzą różne sytuacje. Korzyść jest największa tam, gdzie zadanie jest nowe, a programista ma mniej doświadczenia lub mniejszą znajomość kodu. Maleje, a w skrajnym przypadku staje się ujemna, gdy doświadczona osoba pracuje w dużej, dobrze sobie znanej bazie kodu o wysokich wymaganiach jakościowych. W takim kontekście czas poświęcony na formułowanie poleceń, oczekiwanie i weryfikację odpowiedzi może przewyższyć czas zaoszczędzony na pisaniu. Autorzy badania METR zastrzegają przy tym, że wyniku nie należy uogólniać na wszystkie zastosowania ani na przyszłe generacje narzędzi [5].

4. WYNIKI NA POZIOMIE ORGANIZACJI

Szybsze pisanie kodu przez pojedynczą osobę nie musi oznaczać szybszego i stabilniejszego dostarczania oprogramowania przez zespół. Raport DORA z 2024 r. oszacował, że wzrost adopcji AI o 25% wiązał się ze spadkiem przepustowości dostarczania o 1,5% i spadkiem stabilności dostarczania o 7,2% [6]. Jako prawdopodobne wyjaśnienie autorzy wskazali większy rozmiar zmian: kod generowany łatwo powstaje w dużych porcjach, a duże zmiany są trudniejsze do przeglądu i częściej powodują awarie.

W edycji z 2025 r. obraz jest częściowo inny. Adopcja AI wiąże się tam z wyższą przepustowością dostarczania, natomiast stabilność pozostaje słabym punktem. Główna teza raportu brzmi, że AI działa jak wzmacniacz: poprawia wyniki organizacji o dobrych praktykach inżynierskich i pogłębia problemy tych, które ich nie mają [2].

Wyniki DORA pochodzą z badań ankietowych i opisują zależności statystyczne, a nie związki przyczynowe. Są jednak spójne z wnioskiem płynącym z eksperymentów: wąskim gardłem przestaje być pisanie kodu, a stają się nim przegląd, testowanie i wdrażanie.

5. BEZPIECZEŃSTWO GENEROWANEGO KODU

Pearce i współautorzy przygotowali 89 scenariuszy odpowiadających znanym klasom podatności i wygenerowali dla nich 1689 programów za pomocą asystenta GitHub Copilot. Około 40% z nich zawierało podatności[7]. Badanie dotyczyło wczesnej wersji narzędzia i celowo dobranych scenariuszy ryzykownych, więc odsetka tego nie należy przenosić wprost na dzisiejsze modele. Pokazuje ono jednak, że model uczony na publicznym kodzie odtwarza także jego wady.

Perry i współautorzy zbadali, jak z asystentem pracują ludzie. W ich eksperymencie osoby mające dostęp do asystenta AI pisały kod istotnie mniej bezpieczny niż grupa kontrolna, a jednocześnie częściej uważały, że ich kod jest bezpieczny[8]. Uczestnicy, którzy mniej ufali narzędziu i aktywniej modyfikowali polecenia oraz odpowiedzi, tworzyli kod z mniejszą liczbą podatności.

Wynik ten koresponduje z obserwacją z badania METR. W obu przypadkach narzędzie zwiększało pewność użytkownika bardziej niż faktyczną jakość rezultatu. Z punktu widzenia organizacji to ważniejsze ryzyko niż sam odsetek błędów: błąd, którego autor się spodziewa, zostanie wychwycony w przeglądzie, a błąd w kodzie uznanym za poprawny trafi na produkcję.

6. IMPLIKACJE DLA ZESPOŁÓW I SZKOLEŃ

Z przedstawionych badań wynikają cztery zalecenia praktyczne.

Mierzyć, a nie pytać. Skoro samoocena przyspieszenia może różnić się od pomiaru nawet znakiem, decyzji o wdrożeniu nie należy opierać na ankietach satysfakcji. Właściwymi miarami są czas realizacji zmian, częstość wdrożeń, odsetek zmian powodujących awarie i czas przywracania działania.

Utrzymywać małe zmiany. Łatwość generowania kodu sprzyja dużym porcjom zmian. Limit rozmiaru zmiany i obowiązkowy przegląd przez człowieka ograniczają ryzyko wskazane w raportach DORA.

Wzmocnić testy i przegląd bezpieczeństwa. Automatyczna analiza statyczna, testy i przegląd kodu powinny obejmować kod generowany na takich samych zasadach jak kod pisany ręcznie. Dotyczy to także zgodności z wymaganiami dostępności, których asystent nie zapewnia samoczynnie.

Szkolić z weryfikacji, nie tylko z obsługi. Program szkolenia powinien obejmować formułowanie poleceń, ale przede wszystkim krytyczną ocenę odpowiedzi: typowe klasy błędów, podatności i sytuacje, w których użycie narzędzia się nie opłaca.

Ostatnie zalecenie ma także podstawę prawną. Artykuł 4 rozporządzenia (UE) 2024/1689, czyli aktu w sprawie sztucznej inteligencji, zobowiązuje dostawców i podmioty stosujące systemy AI do podjęcia środków zapewniających odpowiedni poziom kompetencji w zakresie AI wśród personelu

zajmującego się działaniem i wykorzystaniem tych systemów. Przepis ten stosuje się od 2 lutego 2025 r. [9]

7. OGRANICZENIA PRZEGLĄDU

Przegląd nie jest systematycznym przeglądem literatury. Obejmuje wybrane, najczęściej cytowane badania o jawnej metodyce. Część z nich to preprinty, które nie przeszły recenzji naukowej. Badania eksperymentalne dotyczą konkretnych narzędzi i wersji modeli, które szybko się zmieniają, więc wartości liczbowe opisują stan z dnia badania. Dane ankietowe Stack Overflow i DORA pochodzą z prób, które nie są reprezentatywne dla całej populacji programistów. Wyniki raportów DORA przytoczono za ich oficjalnymi omówieniami.

8. WNIOSKI

Wpływ asystentów AI na produktywność zależy od kontekstu. Jest wyraźnie dodatni w nowych, dobrze zdefiniowanych zadaniach i u mniej doświadczonych osób, a może być ujemny u doświadczonych programistów pracujących w dojrzałym kodzie.

Subiektywne poczucie przyspieszenia nie jest wiarygodną miarą. W badaniu METR różniło się od pomiaru kierunkiem.

Na poziomie organizacji korzyści zależą od jakości procesu wytwarzania. Bez małych zmian, testów i przeglądu rośnie niestabilność dostarczania.

Generowany kod może zawierać podatności, a użytkownicy asystentów mają skłonność do przeceniania jego bezpieczeństwa.

Szkolenie zespołów powinno koncentrować się na weryfikacji wyników, co jest zgodne z obowiązkiem zapewnienia kompetencji w zakresie AI wynikającym z prawa Unii.

PRZYPISY

Stack Overflow, 2025 Developer Survey, sekcja AI, 2025.

DORA, Google Cloud, State of AI-assisted Software Development, 2025.

Peng S., Kalliamvakou E., Cihon P., Demirer M., The Impact of AI on Developer Productivity: Evidence from GitHub Copilot, arXiv:2302.06590, 2023.

Cui Z. K., Demirer M., Jaffe S., Musolff L., Peng S., Salz T., The Effects of Generative AI on High-Skilled Work: Evidence from Three Field Experiments with Software Developers, „Management Science”, DOI 10.1287/mnsc.2025.00535; wersja robocza: SSRN 4945566, 2024.

Becker J., Rush N., Barnes E., Rein D., Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity, arXiv:2507.09089, 2025.

DORA, Google Cloud, Accelerate State of DevOps Report 2024, 2024.

Pearce H., Ahmad B., Tan B., Dolan-Gavitt B., Karri R., Asleep at the Keyboard? Assessing the Security of GitHub Copilot's Code Contributions, IEEE Symposium on Security and Privacy, 2022, arXiv:2108.09293.

Perry N., Srivastava M., Kumar D., Boneh D., Do Users Write More Insecure Code with AI Assistants?, ACM CCS 2023, arXiv:2211.03622.

Rozporządzenie Parlamentu Europejskiego i Rady (UE) 2024/1689 z 13 czerwca 2024 r. w sprawie ustanowienia zharmonizowanych przepisów dotyczących sztucznej inteligencji, art. 4 i art. 113.

BIBLIOGRAFIA

Becker J., Rush N., Barnes E., Rein D., Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity, 2025. arxiv.org/abs/2507.09089

Cui Z. K., Demirer M., Jaffe S., Musolff L., Peng S., Salz T., The Effects of Generative AI on High-Skilled Work: Evidence from Three Field Experiments with Software Developers, „Management Science". doi.org/10.1287/mnsc.2025.00535; wersja robocza: papers.ssrn.com/abstract=4945566

DORA, Google Cloud, Accelerate State of DevOps Report 2024. dora.dev/research/2024/dora-report

DORA, Google Cloud, State of AI-assisted Software Development, 2025. dora.dev/research/2025/dora-report

Pearce H., Ahmad B., Tan B., Dolan-Gavitt B., Karri R., Asleep at the Keyboard? Assessing the Security of GitHub Copilot's Code Contributions, 2022. arxiv.org/abs/2108.09293

Peng S., Kalliamvakou E., Cihon P., Demirer M., The Impact of AI on Developer Productivity: Evidence from GitHub Copilot, 2023. arxiv.org/abs/2302.06590

Perry N., Srivastava M., Kumar D., Boneh D., Do Users Write More Insecure Code with AI Assistants?, 2023. arxiv.org/abs/2211.03622

Rozporządzenie Parlamentu Europejskiego i Rady (UE) 2024/1689 z 13 czerwca 2024 r. (akt w sprawie sztucznej inteligencji). eur-lex.europa.eu/eli/reg/2024/1689/oj

Stack Overflow, 2025 Developer Survey, sekcja AI. survey.stackoverflow.co/2025/ai

SŁOWNIK

Asystent kodu

Narzędzie oparte na modelu językowym, które podpowiada, generuje lub zmienia kod programu na podstawie polecenia programisty i kontekstu projektu.

Duży model językowy (LLM)

Model sztucznej inteligencji wytrenowany na bardzo dużych zbiorach tekstu, który generuje tekst lub kod w odpowiedzi na polecenie. Skrót od angielskiego large language model.

Eksperyment randomizowany

Badanie, w którym uczestników lub zadania przydziela się losowo do grupy korzystającej z badanego rozwiązania i do grupy kontrolnej. Losowy przydział pozwala wnioskować o przyczynie różnic.

Preprint

Wstępna wersja pracy naukowej udostępniona publicznie przed recenzją i publikacją w czasopiśmie.

DORA

Program badawczy prowadzony w ramach Google Cloud, który co roku bada praktyki i wyniki zespołów wytwarzających oprogramowanie. Skrót od DevOps Research and Assessment.

Przepustowość i stabilność dostarczania

Dwie miary pracy zespołu programistycznego. Przepustowość opisuje, jak szybko i często zmiany trafiają do użytkowników, a stabilność, jak często te zmiany powodują awarie.

Podatność

Błąd w oprogramowaniu, który można wykorzystać do naruszenia bezpieczeństwa systemu lub danych.

AI Act

Rozporządzenie Parlamentu Europejskiego i Rady (UE) 2024/1689, czyli unijny akt w sprawie sztucznej inteligencji. Określa obowiązki dostawców i podmiotów stosujących systemy AI.

Kompetencje w zakresie AI

Umiejętności, wiedza i zrozumienie pozwalające świadomie korzystać z systemów AI oraz oceniać związane z nimi szanse i ryzyka. Pojęcie zdefiniowane w akcie w sprawie sztucznej inteligencji.

JAK CYTOWAĆ

Załucki T., *Asystenci AI w wytwarzaniu oprogramowania. Przegląd badań nad produktywnością, jakością i bezpieczeństwem kodu*, 2026, dostęp online.

Tekst ma charakter przeglądowy i informacyjny. Nie stanowi porady prawnej. Stan prawny i stan badań na dzień publikacji.