O'REILLY®

# Inżynieria oprogramowania według Google

## Czego warto się nauczyć o tworzeniu oprogramowania

Titus Winters,
Tom Manshreck, Hyrum Wright

Helion

Tytuł oryginału: Software Engineering at Google: Lessons Learned from Programming Over Time

Tłumaczenie: Piotr Pilch

ISBN: 978-83-283-9971-6



Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
*https://helion.pl/user/opinie/iogoog*
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *https://helion.pl* (księgarnia internetowa, katalog książek)

Printed in Poland.

# Spis treści

ROZDZIAŁ 3.

# Dzielenie się wiedzą

*Autorzy: Nina Chen i Mark Barolak*
*Zredagowane przez Rionę MacNamara*

Twoja firma rozumie Twój obszar problemów lepiej niż jakaś losowo wybrana osoba w internecie. Twoja firma powinna być w stanie udzielić odpowiedzi na większość pytań, które sama formułuje. Aby to osiągnąć, niezbędni są zarówno eksperci, którzy znają odpowiedzi na te pytania, *jak i* mechanizmy dystrybucji ich wiedzy. Tą kwestią zajmiemy się w niniejszym rozdziale. Wśród tych mechanizmów są zarówno zupełnie proste (zadaj pytania i zanotuj to, co wiesz), jak i te cechujące się znacznie bardziej złożoną strukturą, takie jak kursy i zajęcia. Najważniejsze jest jednak to, że Twoja firma wymaga *kultury uczenia się*, z czym wiąże się konieczność zapewnienia bezpieczeństwa psychologicznego powalającego ludziom przyznać się do braku wiedzy.

## Wyzwania związane z uczeniem się

Dzielenie się wiedzą w obrębie organizacji nie jest łatwym zadaniem. Bez solidnej kultury uczenia się mogą pojawić się trudności. Firma Google miała do czynienia z kilkoma takimi wyzwaniami, a zwłaszcza wtedy, gdy zwiększała skalę działania. Oto one:

*Brak bezpieczeństwa psychologicznego*

Dotyczy to środowiska, w którym ludzie obawiają się podejmowania ryzyka lub popełniania błędów przy innych osobach, ponieważ odczuwają lęk przed tym, że spotka ich za to kara. Często jest to manifestowane jako kultura strachu lub tendencja do unikania transparentności.

*„Wyspy" informacji*

Terminem tym określa się fragmentację wiedzy mającą miejsce w różnych częściach organizacji, które nie komunikują się ze sobą lub nie korzystają ze wspólnych zasobów. W takim środowisku każda grupa opracowuje swój własny styl realizowania różnych zadań[1]. Prowadzi to często do następujących rzeczy:

[1] Inaczej mówiąc, zamiast określenia jednego maksimum globalnego, dysponujemy grupą maksimów lokalnych (*https://pl.wikipedia.org/wiki/Ekstremum_funkcji*).

*Fragmentacja informacji*

Każda „wyspa" dysponuje niekompletnym obrazem większej całości.

*Duplikowanie informacji*

W ramach każdej „wyspy" wymyśla się na nowo własną metodę realizowania czegoś.

*Wypaczenie informacji*

W obrębie każdej „wyspy" są stosowane własne metody wykonywania tej samej rzeczy, które mogą powodować wzajemnie konflikt albo nie.

*Pojedynczy punkt awarii*

Jest to wąskie gardło pojawiające się, gdy krytyczne informacje są oferowane przez tylko jedną osobę. Jest to powiązane ze *wskaźnikiem autobusowym* (*https://en.wikipedia.org/wiki/Bus_factor*), który bardziej szczegółowo omówiono w rozdziale 2.

Pojedyncze punkty awarii mogą być wynikiem dobrych intencji: bez trudu można wyrobić w sobie nawyk w rodzaju „pozwól, że zajmę się tym za ciebie". Takie podejście zapewnia jednak optymalizowanie pod kątem krótkoterminowej efektywności („poradzę sobie z tym szybciej") kosztem kiepskiej skalowalności długoterminowej (zespół nigdy nie nauczy się, jak postępować, gdy coś musi zostać zrealizowane, niezależnie od tego, co to będzie). Takie nastawienie prowadzi też zwykle do *kompetencji typu „wszystko albo nic"*.

*Kompetencje typu „wszystko albo nic"*

Grupa ludzi dzieląca się na osoby znające się „na wszystkim" i nowicjuszy, w której niewiele osób stanowi kompromis dla obu tych podgrup. Tego rodzaju problem nasila się sam, jeśli eksperci zawsze robią wszystko sami i nie poświęcają czasu na przygotowanie nowych ekspertów w ramach procesu doradztwa lub korzystania z dokumentacji. W tym wariancie wiedza i zakres odpowiedzialności w dalszym ciągu są akumulowane w kręgu osób, które już są ekspertami, a nowi członkowie zespołu lub nowicjusze są pozostawieni sami sobie i wolniej robią postępy.

*Papugowanie*

Naśladowanie bez zrozumienia. Charakteryzuje się to zwykle bezmyślnym kopiowaniem wzorców lub kodu bez zrozumienia jego przeznaczenia. Często odbywa się przy założeniu, że wspomniany kod jest niezbędny z nieznanych powodów.

*„Nawiedzone cmentarze"*

Są to miejsca, często w kodzie, w przypadku których ludzie unikają sprawdzania lub modyfikowania, gdyż obawiają się, że coś może się nie powieść. W przeciwieństwie do wcześniej wspomnianego *papugowania* „nawiedzone cmentarze" są powiązane z osobami unikającymi podejmowania działania ze strachu lub z powodu przesądów.

W pozostałej części rozdziału zagłębimy się w strategie, które okazały się skuteczne w działach inżynieryjnych firmy Google mających do czynienia z przedstawionymi wyzwaniami.

# Filozofia

Inżynieria oprogramowania może być definiowana jako proces projektowania przez wiele osób programów z wieloma wersjami[2]. Ludzie znajdują się w centrum inżynierii oprogramowania: kod stanowi istotny rezultat, lecz jest jedynie niewielką częścią procesu tworzenia produktu. Najważniejsze jest to, że kod nie powstaje spontanicznie z niczego, tak samo jak kompetencje. Każdy ekspert był kiedyś nowicjuszem: sukces organizacji zależy od rozwoju jej pracowników i inwestowania w nich.

Bezpośrednie i dopasowane rady eksperta są zawsze bezcenne. Poszczególni członkowie zespołu mogą mieć różne zakresy kompetencji, dlatego też będą się zmieniać najlepsze osoby w zespole, którym zostanie zadane dane pytanie. Jeśli jednak ekspert uda się na wakacje lub przejdzie do innego zespołu, zespół może „pozostać na lodzie". Choć jedna osoba może być w stanie zapewnić pomoc w dopasowanej formie wielu osobom, takie rozwiązanie nie pozwala na skalowanie i jest ograniczone do niewielkiej liczby „wielu osób".

Z kolei udokumentowana wiedza może być lepiej skalowana nie tylko na poziomie zespołu, ale całej organizacji. Mechanizmy takie jak zespołowe strony *wiki* umożliwiają wielu twórcom dzielenie się swoją wiedzą z większą grupą. Ale nawet pomimo tego, że zapisana dokumentacja jest bardziej skalowalna niż rozmowy dwóch osób, takiej skalowalności towarzyszą pewne kompromisy: dokumentacja może być bardziej ogólna i mniej odpowiednia do wykorzystania w sytuacjach, z jakimi mają do czynienia poszczególne osoby uczące się. Ponadto z dokumentacją jest związany dodatkowy koszt utrzymania niezbędnego do zachowania odpowiedniości i aktualności informacji wraz z upływem czasu.

Wiedza w obrębie grupy istnieje w obszarze znajdującym się między tym, co wiedzą poszczególni członkowie zespołu, a tym, co zostało udokumentowane. Prawdziwi eksperci dysponują wiedzą o rzeczach, które nie zostały udokumentowane. Jeśli taka wiedza zostanie uwzględniona w dokumentacji i będzie utrzymywana, stanie się już dostępna nie tylko dla kogoś, kto może dzisiaj spotkać się bezpośrednio z ekspertem na indywidualną rozmowę, ale dla każdego z możliwością znalezienia i wyświetlenia dokumentacji.

A zatem w magicznym świecie, w którym wszystko jest zawsze idealnie i od razu dokumentowane, nie mielibyśmy już więcej potrzeby konsultowania się z kimkolwiek. Czyż nie? Nie do końca. Udokumentowana wiedza oferuje korzyści związane ze skalowaniem, ale to samo zapewnia ukierunkowana pomoc udzielona przez człowieka. Prawdziwy ekspert potrafi ustalić koszt swojej wiedzy. Ma on możliwość oceny, jakie informacje są wykorzystywane w konkretnym przypadku danej osoby, a także ustalenia, czy dokumentacja nadal jest odpowiednia. Ekspert wie również, gdzie ją znaleźć. Jeśli okaże się, że nie wie, gdzie znaleźć odpowiedzi, może wiedzieć, kto to umożliwi.

Wiedza w obrębie grupy i wiedza udokumentowana uzupełniają się wzajemnie. Nawet członkowie idealnego zespołu eksperckiego dysponujący perfekcyjną dokumentacją muszą się komunikować ze sobą, koordynować działania z innymi zespołami oraz dopasowywać swoje strategie z upływem czasu. Żadna pojedyncza metoda udostępniania wiedzy nie będzie właściwym rozwiązaniem dla każdego typu

[2] David Lorge Parnas, *Software Engineering: Multi-person Development of Multi-version Programs* (Springer-Verlag Berlin, Heidelberg 2011).

uczenia się. Ponadto szczegóły odpowiedniej kombinacji metod będą się prawdopodobnie zmieniać dla poszczególnych organizacji. Wiedza instytucjonalna jest rozwijana z czasem, a metody dzielenia się wiedzą, które najlepiej sprawdzają się w przypadku Twojej firmy, będą raczej się zmieniać wraz z jej powiększaniem się. Prowadź szkolenia, koncentruj się na uczeniu i rozwoju, a także twórz własną grupę ekspertów; nie istnieje coś takiego jak zbyt duża wiedza inżynieryjna.

# Przygotowanie: bezpieczeństwo psychologiczne

Bezpieczeństwo psychologiczne ma kluczowe znaczenie w odniesieniu do promowania środowiska służącego do uczenia się.

Aby się uczyć, musisz najpierw uświadomić sobie, że istnieją rzeczy, których nie rozumiesz. Taka szczerość powinna być przez nas mile widziana (*https://xkcd.com/1053/*), a nie karana (firma Google radzi sobie z tym naprawdę dobrze, ale czasami inżynierowie mają opór, aby przyznać, że czegoś nie rozumieją).

Ogromną część procesu uczenia się stanowi możliwość wypróbowywania różnych rzeczy i poczucia komfortu, gdy coś się nie powiedzie. W zdrowym środowisku ludzie z zadowoleniem zadają pytania, popełniają pomyłki i uczą się nowych rzeczy. Jest to podstawowe oczekiwanie w przypadku wszystkich zespołów w firmie Google. I faktycznie, nasze badania (*https://rework.withgoogle.com/blog/five-keys-to-a-successful-google-team/*) pokazały, że bezpieczeństwo psychologiczne to najważniejsza część skutecznego zespołu.

## Doradztwo

W firmie Google próbujemy zapewnić odpowiednie warunki od razu, jak tylko dołączy do niej nowy inżynier („Noogler"). Istotnym sposobem budowania bezpieczeństwa psychologicznego jest przydzielanie takim nowym inżynierom mentora, czyli kogoś, kto *nie* jest członkiem ich zespołu, menedżerem lub kierownikiem projektu. Zakres odpowiedzialności mentora wyraźnie obejmuje udzielanie odpowiedzi na pytania i pomaganie nowym pracownikom w rozwoju. Łatwiejsze staje się dla początkujących osób poproszenie o pomoc oficjalnie przydzielonego mentora, a ponadto oznacza to, że nie muszą one martwić się tym, że zajmują zbyt wiele czasu swoim współpracownikom.

Mentor to wolontariusz, który pracuje w firmie Google więcej niż rok, a ponadto ma możliwość oferowania rad na różne tematy, począwszy od korzystania z infrastruktury firmy, a skończywszy na radzeniu sobie z kwestią kultury pracy. Co najważniejsze, mentor ma pełnić rolę osoby oferującej wsparcie, z którą jego podopieczny może porozmawiać, gdy nie wie, kogo jeszcze może poprosić o pomoc. Taki mentor nie jest członkiem tego samego zespołu, do którego należy podopieczny. Dzięki temu w złożonych sytuacjach pytający może poczuć się bardziej komfortowo, gdy musi zwrócić się o pomoc.

Doradztwo formalizuje i ułatwia uczenie się, które samo jednak jest ciągłym procesem. Zawsze pojawią się dla współpracowników możliwości nauczenia się czegoś od siebie, niezależnie od tego, czy dotyczy to nowego pracownika dołączającego do firmy, czy doświadczonego inżyniera poznającego nową

technologię. W przypadku właściwie prowadzonego zespołu jego członkowie będą otwarci nie tylko na udzielanie odpowiedzi na pytania, ale też na *zadawanie* ich: członkowie zespołu będą ujawniać, że czegoś nie wiedzą, co pozwoli im na uczenie się od siebie nawzajem.

## Bezpieczeństwo psychologiczne w dużych grupach

Poproszenie o pomoc znajdującego się w pobliżu członka zespołu jest łatwiejsze niż kierowanie się z taką prośbą do dużej grupy złożonej głównie z nieznajomych. Jak jednak zaprezentowano, rozwiązania polegające na bezpośrednim kontakcie dwóch osób nie skalują się dobrze. Rozwiązania grupowe są bardziej skalowalne, ale też powodują większe obawy. Onieśmielające dla początkujących może być formułowanie pytania i zadawanie go dużej grupie, gdy wiedzą, że ich pytanie może być przechowywane przez wiele lat. Potrzeba bezpieczeństwa psychologicznego jest wzmacniana w dużych grupach. Każdy członek grupy ma rolę do odegrania w procesie tworzenia i utrzymywania bezpiecznego środowiska zapewniającego, że początkujący mogą z przekonaniem zadawać pytania. Ponadto dobrze zapowiadający się eksperci czują się upoważnieni do udzielenia pomocy takim początkującym bez obawy, że ich odpowiedzi padną ofiarą ataku ugruntowanych ekspertów.

Najważniejszym sposobem osiągnięcia takiego bezpiecznego i przyjaznego środowiska jest zadbanie o to, aby interakcje w grupie cechowały się współpracą, a nie antagonizmami. W tabeli 3.1 zestawiono kilka przykładów zalecanych wzorców (oraz odpowiadających im antywzorców) dotyczących interakcji w grupie.

*Tabela 3.1. Wzorce interakcji w grupie*

| Zalecane wzorce (oparte na współpracy) | Antywzorce (z antagonizmami) |
|---|---|
| Proste pytania lub błędy są obsługiwane przez kierowanie w odpowiednie miejsce. | Dokonuje się wyboru spośród prostych pytań lub błędów, a osobie, która zadała pytanie, udziela się reprymendy. |
| Wyjaśnienia są przekazywane z zamiarem udzielenia pomocy osobie zadającej pytanie w trakcie nauki. | Wyjaśnienia są udzielane z zamiarem popisania się własną wiedzą. |
| Odpowiedzi są udzielane w sposób przyjazny, pomocny i cierpliwy. | Odpowiedzi są protekcjonalne, złośliwe i niekonstruktywne. |
| Interakcje mają postać wspólnych dyskusji mających na celu znalezienie rozwiązań. | Interakcje polegają na wymianie argumentów uwzględniających kwestie „wygranych" i „przegranych". |

Takie antywzorce mogą zostać spowodowane nieumyślnie: ktoś może próbować być pomocny, ale przypadkiem okazuje się nieprzyjazny i protekcjonalny. Za pomocne uważamy tutaj reguły społeczne podane na stronie Recurse Center (*https://www.recurse.com/manual#sub-sec-social-rules*). Oto one:

*Żadnego udawanego zaskoczenia („Co takiego?! Nie mogę uwierzyć, że nie wiesz, czym jest stos!")*

Udawane zaskoczenie to bariera związana z bezpieczeństwem psychologicznym, która sprawia, że członkowie grupy obawiają się przyznać do braku wiedzy.

*Żadnych „cóż, właściwie to"*

Dotyczy to pedantycznych korekt, które zwykle mają więcej wspólnego z populizmem niż z dokładnością.

*Żadnego „sterowania z tylnego siedzenia”*

Mowa o przerywaniu trwającej dyskusji w celu przedstawienia opinii bez zaangażowania się w rozmowę.

*Żadnych subtelnych „-izmów” („To takie proste, że moja babcia mogłaby to zrobić!”)*

Drobne oznaki uprzedzeń (rasizm, ageizm, homofobia), które mogą spowodować, że poszczególne osoby mogą poczuć się niemile widziane, nieszanowane lub zagrożone.

# Poszerzanie własnej wiedzy

Dzielenie się wiedzą zaczyna się od samego siebie. Ważne jest zdanie sobie sprawy z tego, że zawsze jest coś do nauczenia. Zamieszczone dalej wytyczne pozwalają poszerzyć osobistą wiedzę.

## Zadawaj pytania

Jeśli z treści tego rozdziału uzyskasz tylko jedną rzecz, jest ona następująca: zawsze ucz się i zadawaj pytania.

Nowych pracowników informujemy o tym, że wdrażanie może zająć około 6 miesięcy. Taki spory zakres czasu jest niezbędny do zaznajomienia się z dużą i złożoną infrastrukturą firmy Google, ale też wzmacnia koncepcję, że uczenie się to ciągły i iteracyjny proces. Jednym z największych błędów popełnianych przez początkujących jest to, że nie proszą o pomoc, gdy nie mogą sobie z czymś poradzić. Można ulec pokusie, aby w pojedynkę borykać się z jakąś trudnością, lub można mieć obawę o to, że nasze pytania są „zbyt proste”. Myślisz sobie: „Po prostu muszę bardziej się postarać, zanim poproszę kogokolwiek o pomoc”. Nie wpadnij w taką pułapkę! Twoi współpracownicy to często najlepsze źródło informacji — wykorzystaj ten cenny zasób.

Nie będzie żadnego magicznego dnia, gdy nagle będziesz wiedział dokładnie, co zrobić w każdej sytuacji. Zawsze można się nauczyć czegoś jeszcze. Inżynierowie pracujący od wielu lat w firmie Google w dalszym ciągu mają do czynienia z dziedzinami, w przypadku których nie czują, że wiedzą, co robią. Nie stanowi to jednak problemu! Nie obawiaj się powiedzieć: „Nie wiem, co to jest. Czy mógłbyś to objaśnić?”. Potraktuj kwestię nieznajomości rzeczy bardziej jako obszar możliwości niż coś, czego należy się obawiać[3].

Nie ma znaczenia, czy jesteś nowym członkiem zespołu, czy doświadczonym liderem — zawsze powinno się być w środowisku, gdzie można się czegoś jeszcze nauczyć. W przeciwnym razie padniesz ofiarą stagnacji (powinno się wtedy poszukać nowego środowiska).

Dla osób będących liderami szczególnie ważne jest modelowanie takiego zachowania: istotne jest, aby omyłkowo nie utożsamiać „stażu pracy” ze „znajomością wszystkiego”. Tak naprawdę im więcej wiesz, tym bardziej jesteś świadom, czego nie wiesz (*https://phdcomics.com/comics.php?f=1056*).

[3] Syndrom oszusta (*https://pl.wikipedia.org/wiki/Syndrom_oszusta*) występuje często wśród ludzi sukcesu. Pracownicy firmy Google nie są tu wyjątkiem. Okazuje się, że większość autorów tej książki cierpi na ten syndrom. Potwierdzamy, że obawa przed niepowodzeniem może być czymś trudnym dla osób z syndromem oszusta, a ponadto może nasilać skłonności do rozwijania zakresu zainteresowań.

Zadawanie pytań w otwarty sposób[4] lub prezentowanie luk w zasobach wiedzy wzmacnia przekonanie, że inne osoby mogą jak najbardziej postępować w ten sam sposób.

Z perspektywy odbiorców cierpliwość i życzliwość towarzyszące zadawaniu pytań sprzyjają tworzeniu środowiska, w którym ludzie nie boją się szukania pomocy. Ułatwienie pokonania początkowego wahania w kwestii zadania pytania pozwala szybko zapewnić odpowiedni komfort. Trzeba zabiegać o to, aby pytania były zadawane, a ponadto o ułatwienie otrzymania odpowiedzi nawet na „trywialne" pytania. Choć inżynierowie *mogą* prawdopodobnie samodzielnie uzyskać wiedzę dostępną w obrębie grupy, nie są po to, aby pracować w próżni. Ukierunkowana pomoc pozwala inżynierom na szybsze stanie się wydajnymi, co z kolei zwiększa wydajność całego zespołu.

## Zrozum kontekst

Uczenie się nie ogranicza się jedynie do zrozumienia nowych rzeczy. Obejmuje również opracowanie informacji umożliwiających zrozumienie decyzji związanych z projektem i implementacją istniejących rzeczy. Załóżmy, że Twój zespół odziedziczył starszą bazę kodu powiązaną z kluczową częścią infrastruktury, która istniała przez wiele lat. Oryginalnych twórców kodu od dawna nie ma w firmie, a kod jest trudny do zrozumienia. Może pojawić się pokusa, aby ponownie utworzyć kod od początku niż raczej poświęcać czas na zaznajamianie się z istniejącym kodem. Zamiast jednak pomyśleć „nie rozumiem tego" i na tym zakończyć rozważania, warto zagłębić się w to bardziej: jakie pytania powinno się zadać?

Weź pod uwagę zasadę „płotu Chestertona" (*https://en.wikipedia.org/wiki/Wikipedia:Chesterton's_fence*): przed usunięciem lub zmodyfikowaniem czegoś zrozum najpierw, dlaczego to coś istnieje.

> W kwestii reformowania rzeczy, w odróżnieniu od ich deformacji, istnieje jedna zwyczajna i prosta zasada; zasada, która prawdopodobnie będzie nazywana paradoksem. Działa w takim przypadku pewna instytucja lub prawo; powiedzmy, dla uproszczenia, ogrodzenie lub brama wzniesiona po drugiej stronie drogi. Bardziej nowoczesny typ reformatora idzie do niego wesoło i mówi: „Nie widzę celu jego wykorzystania; oczyśćmy to". Na co bardziej inteligentny typ reformatora dobrze odpowie: „Jeśli nie widzisz jego wykorzystania, na pewno nie pozwolę ci go usunąć. Odejdź i pomyśl. Następnie, kiedy będziesz mógł wrócić i powiesz mi, że widzisz, do czego on służy, mogę pozwolić ci go zniszczyć".

Nie oznacza to, że kod nie może być pozbawiony przejrzystości albo że istniejące wzorce projektowe nie mogą być niepoprawne. Inżynierowie mają jednak tendencję do posługiwania się stwierdzeniem „to jest złe!" zdecydowanie szybciej, niż często są ku temu podstawy, zwłaszcza w przypadku kodu, języków lub modelów, z którymi nie są zaznajomieni. Firma Google też nie jest na to odporna. Zidentyfikuj i zrozum kontekst, co szczególnie dotyczy decyzji, które wydają się być nietypowe. Po zaznajomieniu się z kontekstem i przeznaczeniem kodu rozważ to, czy utworzona zmiana nadal ma sens. Jeśli tak, możesz śmiało ją wprowadzić. W przeciwnym razie udokumentuj własne wnioskowanie z myślą o kolejnych osobach czytających dokumentację.

[4] Zajrzyj na stronę *How to ask good questions* (*https://jvns.ca/blog/good-questions/*).

Wiele wytycznych firmy Google dotyczących stylu jawnie uwzględnia kontekst, aby ułatwić osobom czytającym zrozumienie powodów wprowadzenia takich wytycznych, a nie tylko zapamiętanie listy arbitralnie ustalonych reguł. Ujmując to subtelniej, zrozumienie powodów określenia danej wytycznej pozwala twórcom podejmować przemyślane decyzje odnoszące się do tego, kiedy wytyczna nie powinna być stosowana, albo tego, czy wymaga ona zaktualizowania (zajrzyj do rozdziału 8.).

# Skalowanie pytań: zadaj je społeczności

Uzyskiwanie pomocy w ramach rozmowy z drugą osobą cechuje się dużą przepustowością informacji, ale siłą rzeczy jest ograniczone z punktu widzenia skali. Ponadto dla osoby uczącej się może być trudne zapamiętanie każdego szczegółu. Zrób sobie samemu przysługę na przyszłość: gdy uczysz się czegoś w trakcie takiego bezpośredniego spotkania, *rób notatki*.

Są szanse na to, że kolejni początkujący będą mieć takie same pytania, które Ty zadałeś. Zrób przysługę także im i *udostępnij to, co zanotowałeś*.

Choć dzielenie się otrzymanymi odpowiedziami może być przydatne, korzystne jest również szukanie pomocy nie tylko u poszczególnych osób, ale też w obrębie większej społeczności. W niniejszym podrozdziale analizujemy różne formy uczenia się w oparciu o wiedzę społeczności. Każdej z prezentowanych metod (rozmowy grupowe, listy wysyłkowe oraz systemy pytań i odpowiedzi) towarzyszą różne kompromisy. Metody wzajemnie uzupełniają się. Każda z nich umożliwia jednak osobie szukającej informacji uzyskanie pomocy od szerszej społeczności użytkowników i ekspertów, a także zapewnienie, że odpowiedzi są ogólnie dostępne dla obecnych i przyszłych członków tej społeczności.

## Rozmowy grupowe

Gdy masz pytanie, czasami może być trudne otrzymanie pomocy od właściwej osoby. Być może nie jesteś pewien, kto zna odpowiedź, albo osoba, którą chcesz o coś zapytać, jest zajęta. W takich sytuacjach rozmowy grupowe to znakomite rozwiązanie, ponieważ możesz zadać swoje pytanie jednocześnie wielu osobom i szybko przeprowadzić bezpośrednią rozmowę z kimkolwiek, kto jest dostępny. W ramach bonusu inni członkowie rozmowy grupowej mogą dowiedzieć się czegoś z pytania i udzielić na nie odpowiedzi. Wiele form rozmowy grupowej może być objętych automatycznym archiwizowaniem umożliwiającym późniejsze wyszukiwanie.

Rozmowy grupowe są zwykle poświęcone albo zagadnieniom, albo zespołom. Rozmowy dotyczące zagadnień mają przeważnie charakter otwarty, dlatego każdy może do nich dołączyć w celu zadania pytania. Takie konwersacje są zwykle interesujące dla ekspertów i mogą stać się dość obszerne. Dzięki temu odpowiedzi na pytania są zazwyczaj szybko udzielane. Z kolei rozmowy związane z zespołami będą zwykle odbywać się na mniejszą skalę z ich uczestnikami ograniczającymi się do członków zespołu. W efekcie takie rozmowy mogą nie mieć takiego samego zasięgu, jaki mają rozmowy poświęcone zagadnieniom, ale ich mniejsza skala może sprawić, że początkujący poczują się pewniej.

Choć rozmowy grupowe sprawdzają się świetnie w przypadku szybkich pytań, nie zapewniają zbyt wiele struktury, co może utrudnić wyodrębnienie wartościowych informacji z konwersacji, w której nie bierze się aktywnego udziału. Gdy tylko zaistnieje konieczność udostępnienia informacji poza obręb grupy lub po to, aby można było później z nich skorzystać, powinno się utworzyć dokumentację lub wysłać wiadomość pocztową na adres listy wysyłkowej.

## Listy wysyłkowe

Większość zagadnień w firmie Google ma powiązaną z nimi listę wysyłkową w ramach grup Google (topic-users@ lub topic-discuss@), do której każdy w firmie może dołączyć lub wysłać wiadomość. Zadawanie pytania na publicznej liście wysyłkowej przypomina w dużym stopniu formułowanie pytania podczas rozmowy grupowej: pytanie trafia do wielu osób, które mogą potencjalnie udzielić na nie odpowiedzi, a każda osoba obserwująca listę może dzięki pytaniu czegoś się dowiedzieć. W przeciwieństwie jednak do rozmów grupowych publiczne listy wysyłkowe mogą być z łatwością udostępniane szerszemu gronu odbiorców: listy są w postaci pakietów umieszczane w archiwach umożliwiających wyszukiwanie, a wątki wiadomości pocztowych zapewniają bardziej rozbudowaną strukturę niż rozmowy grupowe. W firmie Google listy wysyłkowe są też indeksowane i mogą być wykrywane przez firmową, intranetową wyszukiwarkę Moma.

Po znalezieniu odpowiedzi na pytanie, które zadałeś na liście wysyłkowej, może pojawić się pokusa, aby od razu zająć się własną pracą. Nie rób tego! Nigdy nie wiadomo, kiedy ktoś będzie potrzebować tych samych informacji w przyszłości (*https://xkcd.com/979/*), dlatego najlepszą praktyką jest zamieszczenie odpowiedzi ponownie na liście wysyłkowej.

Listy wysyłkowe nie są pozbawione kompromisów. Dobrze nadają się w przypadku złożonych pytań wymagających sporego kontekstu, ale okazują się niezręczne przy szybkiej wymianie informacji, gdzie świetnie sprawdzają się rozmowy grupowe. Wątek dotyczący konkretnego problemu jest przeważnie najbardziej przydatny, gdy jest aktywny. Archiwa wiadomości pocztowych są trwałe i może być trudne określenie, czy odpowiedź znaleziona w starym wątku dyskusji nadal jest właściwa w odniesieniu do obecnej sytuacji. Ponadto stosunek sygnału do szumu może mieć mniejszą wartość niż dla innych nośników informacji, takich jak formalna dokumentacja, ponieważ problem, jaki ktoś ma w związku z konkretnym przepływem informacji, może nie mieć zastosowania w Twoim przypadku.

### Poczta elektroniczna w firmie Google

Owiana złą sławą kultura pracy w firmie Google bazuje przede wszystkim na poczcie elektronicznej i dużej ilości wysyłanych wiadomości pocztowych. Inżynierowie firmy otrzymują codziennie setki wiadomości (jeśli nie więcej) cechujących się różnym stopniem możliwości podejmowania działań. Nowi pracownicy firmy Google mogą poświęcić wiele dni na samo konfigurowanie filtrów wiadomości pocztowych, aby poradzić sobie z ilością powiadomień przychodzących z grup, do których zostali automatycznie zapisani. Niektórzy po prostu się poddają i nie próbują nadążać za przepływem informacji. Niektóre grupy dla każdej dyskusji domyślnie włączają funkcję tworzenia kopii wiadomości z dużych list wysyłkowych bez podejmowania próby kierowania informacji do osób, które prawdopodobnie będą szczególnie nimi zainteresowane. W efekcie stosunek sygnału do szumu może stać się rzeczywistym problemem.

Domyślnie firma Google dąży przeważnie do przepływów informacji opartych na poczcie elektronicznej. Niekoniecznie wynika to z tego, że poczta jest lepszym nośnikiem niż inne formy komunikacji (często tak nie jest), lecz raczej z tego, że właśnie do takiego rozwiązania dostosowano naszą kulturę pracy. Pamiętaj o tym, gdy Twoja firma będzie zastanawiać się nad tym, jakie formy komunikacji pobudzić lub objąć zakresem inwestycji.

## YAQS — platforma pytań i odpowiedzi

YAQS (*Yet Another Question System*) to wewnętrzna wersja serwisu internetowego Stack Overflow (*https://pl.wikipedia.org/wiki/Stack_Overflow*) używana przez firmę Google. Ułatwia jej pracownikom odnoszenie się do istniejącego kodu lub będącego w trakcie tworzenia, a także dyskutowanie na temat poufnych informacji.

Podobnie do serwisu Stack Overflow, platforma YAQS oferuje wiele tych samych korzyści co listy wysyłkowe i zapewnia udoskonalenia: odpowiedzi oznaczone jako pomocne są promowane w interfejsie użytkownika, a ponadto użytkownicy mogą edytować pytania i odpowiedzi tak, aby pozostały dokładne i przydatne po zmodyfikowaniu kodu i faktów. W rezultacie niektóre listy wysyłkowe zostały zastąpione przez platformę YAQS, natomiast pozostałe przeobraziły się w bardziej ogólne listy dyskusyjne, w przypadku których mniejszą uwagę kładzie się na kwestię rozwiązywania problemów.

# Skalowanie posiadanej wiedzy — zawsze możesz się czegoś nauczyć

Możliwość uczenia nie jest ograniczona do ekspertów. Kompetencje nie są stanem binarnym, w przypadku którego jesteś albo początkującym, albo ekspertem. Kompetencje to wielowymiarowy wektor tego, co wiesz: każdy dysponuje zmiennymi poziomami kompetencji w obrębie różnych dziedzin. Jest to jeden z powodów wyjaśniających, dlaczego różnorodność jest kluczowa do osiągnięcia sukcesu organizacyjnego: różne osoby reprezentują odmienne punkty widzenia i kompetencje (zajrzyj do rozdziału 4.). Inżynierowie firmy Google uczą innych na różne sposoby (np. organizując spotkania w godzinach pracy, oferując konsultacje techniczne, prowadząc zajęcia, tworząc dokumentację i dokonując inspekcji kodu).

## Spotkania w godzinach pracy

Czasami naprawdę ważna jest możliwość porozmawiania z człowiekiem. W takich sytuacjach wyznaczenie w tym celu godzin w trakcie pracy może być dobrym rozwiązaniem. Takie godziny są regularnie zaplanowanym wydarzeniem (zwykle cotygodniowym), w czasie którego jedna lub więcej osób jest dostępnych i może udzielić odpowiedzi na pytania dotyczące konkretnego tematu. Spotkania w godzinach pracy prawie nigdy nie są wybierane jako pierwszy wariant dzielenia się wiedzą: jeśli masz naglące pytanie, kłopotliwa może być konieczność czekania na następną sesję umożliwiającą udzielenie odpowiedzi. Jeśli prowadzisz spotkania w godzinach pracy, pochłania to czas i wymaga regularnego promowania. Zapewniają one jednak ludziom możliwość porozmawiania osobiście z ekspertem. Jest to szczególnie przydatne, gdy problem w dalszym ciągu jest na tyle niejednoznaczny, że inżynier nie wie jeszcze, jakie pytania zadać (na przykład w sytuacji, gdy dopiero rozpoczęto projektowanie nowej usługi) albo czy problem dotyczy czegoś tak specjalistycznego, że zwyczajnie nie ma dostępnej odpowiedniej dokumentacji.

## Konsultacje techniczne i zajęcia

W firmie Google wypracowano solidną kulturę pracy dotyczącą zarówno wewnętrznych, jak i zewnętrznych[5] konsultacji technicznych oraz zajęć. Nasz zespół engEDU (*Engineering Education*) koncentruje się na zapewnianiu edukacji informatycznej wielu różnymi grupom odbiorców, począwszy od inżynierów firmy Google, a skończywszy na studentach z całego świata. Na bardziej podstawowym poziomie nasz program g2g (*Googler2Googler*) umożliwia pracownikom firmy Google rejestrowanie się, aby poprowadzić konsultacje i zajęcia albo wziąć udział w tych organizowanych przez innych pracowników[6]. Program cieszy się dużym powodzeniem. Uczestniczą w nim tysiące pracowników firmy Google, którzy zdobywają wiedzę z różnych dziedzin, począwszy od tych technicznych (np. wektoryzacja w nowoczesnych procesorach), a skończywszy na czysto rozrywkowych (np. taniec swing dla początkujących).

Konsultacje techniczne polegają zwykle na tym, że prelegent prezentuje coś bezpośrednio grupie odbiorców. Z kolei zajęcia mogą uwzględniać wykład, ale często koncentrują się na ćwiczeniach, dlatego wymagają bardziej aktywnego zaangażowania uczestników. Oznacza to, że przygotowywanie i utrzymywanie zajęć prowadzonych przez instruktora jest przeważnie bardziej wymagające i kosztowniejsze niż konsultacje techniczne. W związku z tym zajęcia są przewidziane dla najważniejszych lub najtrudniejszych zagadnień. Gdy zajęcia zostaną już przygotowane, mogą być jednak stosunkowo łatwo skalowane, ponieważ wielu instruktorów może prowadzić zajęcia z wykorzystaniem tych samych materiałów szkoleniowych. Stwierdziliśmy, że zajęcia sprawdzają się zwykle najlepiej, gdy występują następujące okoliczności:

- Zagadnienie jest na tyle skomplikowane, że często stanowi źródło nieporozumień. Przygotowanie zajęć wymaga wiele pracy, dlatego powinny być opracowywane tylko wtedy, gdy służą realizacji konkretnych potrzeb.
- Zagadnienie jest dość niezmienne. Aktualizowanie materiałów na zajęcia wiąże się z mnóstwem pracy, dlatego w sytuacji, gdy zagadnienie szybko się zmienia, bardziej ekonomiczne będą inne formy przekazywania wiedzy.
- Z punktu widzenia zagadnienia korzystna jest dostępność nauczycieli, którzy odpowiadają na pytania i oferują dopasowaną indywidualnie pomoc. Jeśli studenci z łatwością mogą się uczyć bez bezpośredniego wsparcia, efektywniejsze są nośniki informacji umożliwiające korzystanie we własnym zakresie, takie jak dokumentacja lub nagrania. W ramach pewnej liczby zajęć wprowadzających w firmie Google oferowane są też ich wersje do samodzielnej nauki.
- Pojawiło się zapotrzebowanie na zajęcia w stopniu wystarczającym do tego, aby oferować je regularnie. W innej sytuacji potencjalne osoby, które się uczą, zamiast czekać na zajęcia, mogą uzyskać wymagane przez nie informacje za pomocą innych metod. W firmie Google problem ten dotyczy szczególnie niewielkich biur znajdujących się od siebie w dużych odległościach geograficznych.

[5] Można tutaj podać adresy niektórych z nich: *https://talksat.withgoogle.com/* i *https://www.youtube.com/GoogleTechTalks*.

[6] Program g2g opisano szczegółowo w publikacji autorstwa Laszlo Bocka zatytułowanej *Work Rules!: Insights from Inside Google That Will Transform How You Live and Lead* (Twelve Books, New York 2015 r.). W książce uwzględniono opisy różnych aspektów programu, a także wyjaśniono, jak oceniać wpływ, i podano zalecenia dotyczące tego, na czym należy się koncentrować podczas przygotowywania podobnych programów.

## Dokumentacja

Dokumentacja zawiera wiedzę w formie pisemnej. Jej głównym celem jest ułatwienie osobom czytającym nauczenie się czegoś. Nie każda wiedza w postaci pisemnej musi być dokumentacją, choć może być przydatne zaoferowanie jej spisanej na papierze. Możliwe jest na przykład znalezienie rozwiązania problemu w wątku listy wysyłkowej. Podstawowym celem pierwotnego pytania, które pojawiło się w wątku, było jednak poszukiwanie odpowiedzi, a dokumentowanie dyskusji na potrzeby innych osób miało znaczenie jedynie drugorzędne.

W tym punkcie skupiamy się na dostrzeganiu możliwości zaangażowania i tworzenia formalnej dokumentacji, począwszy od drobnych rzeczy, takich jak literówki, a skończywszy na wyzwaniach wymagających większego wysiłku (np. dokumentowanie wiedzy w obrębie grupy).

W rozdziale 10. w szerszym zakresie omówiono kwestię dokumentacji.

### Aktualizowanie dokumentacji

Gdy uczysz się czegoś po raz pierwszy, jest to najlepszy moment na to, aby zastanowić się, czy można w jakiś sposób ulepszyć dokumentację oraz materiały szkoleniowe. Zanim przyswoiłeś sobie nowy proces lub system oraz zrozumiałeś go, być może zapomniałeś, co było trudne, albo jakich prostych kroków nie uwzględniono w dokumentacji wprowadzającej. Jeśli na tym etapie stwierdzisz w dokumentacji błąd lub brak, dokonaj korekty! Pozostaw „obozowisko" czystsze od tego zastanego[7], a ponadto spróbuj sam zaktualizować dokumenty nawet wtedy, gdy dokumentacja należy do innego działu organizacji.

W firmie Google inżynierowie czują się upoważnieni do aktualizowania dokumentacji niezależnie od tego, kto jest jej właścicielem. Często ma to miejsce nawet wtedy, gdy poprawka ogranicza się do usunięcia literówki. Utrzymywanie przez społeczność na takim poziomie zwiększyło się znacząco wraz z wprowadzeniem platformy g3doc[8], która znacznie ułatwiła pracownikom firmy Google identyfikowanie właściciela dokumentacji w celu dokonania przez niego inspekcji proponowanych przez nich poprawek. Platforma pozwala też zachować umożliwiający audyt ślad w postaci historii zmian, który jest taki sam jak w przypadku kodu.

### Tworzenie dokumentacji

Wraz z nabywaniem coraz większej biegłości twórz własną dokumentację i aktualizuj istniejące dokumenty. Jeśli na przykład przygotowujesz nowy przepływ informacji procesu projektowania, dokumentuj jego kroki. Dzięki podaniu odnośników do utworzonej dokumentacji możesz później ułatwić innym osobom postępowanie zgodnie z Twoimi wytycznymi. Jeszcze lepszym rozwiązaniem

[7] Przeczytaj artykuł *The Boy Scout Rule* (*https://biratkirat.medium.com/step-8-the-boy-scout-rule-robert-c-martin-uncle-bob-9ac839778385*) oraz książkę Kevlina Henneya, *97 Things Every Programmer Should Know* (O'Reilly, Boston 2010).

[8] g3doc to skrót od terminu *google3 documentation*. google3 to nazwa aktualnej inkarnacji monolitycznego repozytorium kodu źródłowego firmy Google.

będzie umożliwienie innym samodzielnego znalezienia dokumentów. Jakakolwiek dokumentacja, która nie zapewnia w wystarczającym stopniu możliwości znajdowania lub przeszukiwania jej, równie dobrze może nie istnieć. Jest to kolejny obszar, w którym bryluje platforma g3doc, ponieważ zgodnie z przewidywaniami dokumentacja znajduje się bezpośrednio obok kodu źródłowego, w przeciwieństwie do umieszczonego gdzieś na zewnątrz i niemożliwego do znalezienia dokumentu lub strony internetowej.

I wreszcie zadbaj o dostępność mechanizmu zapewniania informacji zwrotnej. Jeżeli osoby czytające dokumentację nie mogą w prosty i bezpośredni sposób wskazać, że jest ona nieaktualna lub niedokładna, prawdopodobnie nie będzie im zależeć na poinformowaniu o tym kogokolwiek, a kolejny nowicjusz napotka ten sam problem. Ludzie chętniej będą chcieli wprowadzić zmiany, jeśli czują, że ktoś faktycznie dostrzeże i weźmie pod uwagę ich sugestie. W firmie Google błąd w dokumentacji można zgłosić bezpośrednio z poziomu samego dokumentu.

Ponadto pracownicy firmy Google mogą z łatwością pozostawiać komentarze na stronach platformy g3doc. Inni pracownicy mogą je wyświetlać i odpowiadać na nie, ponieważ utworzenie komentarza powoduje automatyczne zgłoszenie błędu właścicielowi dokumentacji. Dzięki temu osoba czytająca ją nie musi stwierdzać, z kim ma się skontaktować.

### Promowanie dokumentacji

Zachęcanie inżynierów do dokumentowania wyników swojej pracy tradycyjnie może być trudne. Tworzenie dokumentacji zajmuje czas i wymaga nakładu, który można przeznaczyć na pisanie kodu. Korzyści wynikające z dokumentowania nie są natychmiastowe i przeważnie dotyczą innych osób. Tego rodzaju asymetryczne kompromisy są dobre z punktu widzenia całej organizacji, gdy pod uwagę weźmie się to, że wiele osób może korzystać z czasu zainwestowanego przez nielicznych. Bez odpowiedniej motywacji zachęcenie do takiego postępowania może być jednak trudne. Część tego rodzaju motywacji strukturalnych omawiamy w zamieszczonym dalej w rozdziale podpunkcie „Motywacje i uznanie”.

Twórca dokumentu może jednak często bezpośrednio skorzystać z faktu pisania dokumentacji. Załóżmy, że członkowie zespołu zawsze proszą Cię o pomoc przy debugowaniu określonego rodzaju błędów produkcyjnych. Dokumentowanie stosowanych procedur wymaga zainwestowania wcześniej czasu, ale po zrobieniu tego możesz w przyszłości zaoszczędzić czas dzięki wskazaniu członkom zespołu dokumentacji i zapewnianiu bezpośredniego wsparcia tylko w razie potrzeby.

Tworzenie dokumentacji ułatwia również skalowanie Twojego zespołu i organizacji. Po pierwsze informacje zawarte w dokumentacji stają się kanoniczne jako materiał referencyjny: członkowie zespołu mogą korzystać z udostępnionego dokumentu, a nawet aktualizować go samemu. Po drugie kanonizacja może wykraczać swoim zasięgiem poza obręb zespołu. Być może niektóre części dokumentacji nie dotyczą wyłącznie konfiguracji stosowanej przez zespół i staną się przydatne dla innych zespołów dążących do rozwiązania podobnych problemów.

## Kod

Na metapoziomie kod *to* wiedza, dlatego samo pisanie kodu może być traktowane jako forma zapisu wiedzy. Choć dzielenie się wiedzą może nie być bezpośrednim celem kodu produkcyjnego, często jest to powstały efekt uboczny, który może zostać ułatwiony dzięki czytelności i przejrzystości kodu.

Dokumentacja kodu to jeden ze sposobów udostępniania wiedzy. Przejrzysta dokumentacja nie tylko zapewnia korzyści użytkownikom biblioteki, ale też osobom, które w przyszłości będą zajmować się jej utrzymywaniem. Podobnie komentarze dotyczące implementacji przekazują wiedzę z upływem czasu: tworzysz je specjalnie z myślą o osobach, które w przyszłości będą czytać dokumentację (włącznie z samym sobą!). Z punktu widzenia kompromisów komentarzom w kodzie towarzyszą te same mankamenty co ogólnej dokumentacji: muszą one być aktywnie utrzymywane albo szybko staną się nieaktualne. Może to zostać potwierdzone przez każdego, kto w ogóle zaznajomił się z treścią komentarza ujawniającą sprzeczność z obecnym przeznaczeniem kodu.

Inspekcje kodu (zajrzyj do rozdziału 9.) to często możliwość nauczenia się czegoś zarówno dla twórców kodu, jak i jego inspektorów. Na przykład sugestia inspektora może sprawić, że twórca kodu skorzysta z nowego wzorca testowania. Ewentualnie inspektor może dowiedzieć się czegoś o nowej bibliotece dzięki użyciu jej przez twórcę w swoim kodzie. Firma Google standaryzuje proces doradzania za pomocą inspekcji kodu obejmującej *proces oceny czytelności* kodu. Szczegółowo zajęto się tą kwestią w analizie przypadku zamieszczonej na końcu rozdziału.

# Skalowanie wiedzy na poziomie organizacji

Zapewnienie, że kompetencje są odpowiednio dostępne w całej organizacji, staje się trudniejsze wraz z jej powiększaniem. Niektóre rzeczy, takie jak kultura pracy, są ważne na każdym etapie rozwoju, natomiast inne kwestie, takie jak ustanawianie kanonicznych źródeł informacji, mogą być korzystniejsze dla bardziej dojrzałych organizacji.

## Doskonalenie kultury dzielenia się wiedzą

Kultura organizacyjna to niepewna kwestia związana z ludźmi, którą wiele firm traktuje jako dodatek. W firmie Google jesteśmy jednak przekonani, że skupienie się najpierw na kwestii kultury i środowiska[9] zapewnia lepsze wyniki niż skupienie się tylko na efekcie (np. kod) działań takiego środowiska.

Dokonywanie poważnych zmian organizacyjnych jest trudne. Na ten temat napisano niezliczoną ilość książek. Nie twierdzimy, że dysponujemy wszystkimi odpowiedziami, ale możemy zaprezentować konkretne kroki, jakie firma Google podjęła w celu zbudowania kultury pracy promującej kwestię uczenia się.

W książce *Work Rules!*[10] bardziej dogłębnie objaśniono zagadnienie kultury pracy w firmie Google.

[9] Laszlo Bock, *Work Rules!: Insights from Inside Google That Will Transform How You Live and Lead* (Twelve Books, New York 2015).

[10] Ibidem.

## Szacunek

Niewłaściwe zachowanie zaledwie kilku osób może sprawić, że nieprzyjazność będzie zauważalna w całym zespole lub społeczności (*https://hbr.org/podcast/2013/01/the-high-cost-of-rudeness-at-w.html*). W takim środowisku początkujący wyrabiają w sobie nawyk zadawania swoich pytań gdzie indziej, a potencjalni nowi eksperci przestają próbować i nie mają możliwości rozwoju. W najgorszych przypadkach w grupie pozostają wyłącznie jej najbardziej „toksyczni" członkowie. Wyprowadzenie grupy z takiego stanu może być trudne.

Proces dzielenia się wiedzą może i powinien przebiegać z życzliwością i szacunkiem. W branży technologicznej tolerowanie „wyjątkowego głupka" lub, co gorsza, darzenie go szacunkiem, jest zarówno wszechobecne, jak i szkodliwe, ale bycie ekspertem i bycie miłym nie wyklucza się wzajemnie. W sekcji dotyczącej przywództwa, stanowiącej część „drabinki" stanowisk w firmie Google związanych z inżynierią oprogramowania, opisano to wyraźnie:

> Choć oczekuje się wyższych poziomów kryterium przywództwa technicznego, nie dotyczy ono wyłącznie problemów technicznych. Liderzy dbają o to, aby otaczały ich bardziej kompetentne osoby, poprawiają poziom bezpieczeństwa psychologicznego zespołu, zapewniają kulturę pracy zespołowej, łagodzą spięcia w zespole, dają przykład uwzględniania kultury pracy i wartości obowiązujących w firmie Google, a także sprawiają, że jest ona bardziej ekscytującym i tętniącym życiem miejscem pracy. Głupki nie są dobrymi liderami.

Takie oczekiwania są modelowane przez doświadczonych przywódców: Urs Hölzle (starszy wiceprezes ds. infrastruktury technicznej) oraz Ben Treynor Sloss (wiceprezes i założyciel serwisu Google SRE) utworzyli regularnie cytowany dokument wewnętrzny („Żadnych głupków"), w którym wyjaśniono, dlaczego pracownicy firmy Google powinni mieć na uwadze zachowanie w pracy pełne szacunku, a także to, jak w związku z tym postępować.

## Motywacje i uznanie

Właściwa kultura pracy musi być aktywnie pielęgnowana, a zachęcanie do dzielenia się wiedzą wymaga zaangażowania na poziomie systemowym w to, aby towarzyszyło temu uznanie i satysfakcja. Częstym błędem popełnianym przez organizacje jest składanie deklaracji bez pokrycia dotyczących poparcia zestawu wartości przy jednoczesnym istnieniu zachowania z aktywnym nagradzaniem, które nie wymusza tych wartości. Ludzie reagują na niebanalne zachęty, dlatego ważne jest potwierdzenie słów czynami w postaci wprowadzenia systemu rekompensat i nagród.

W firmie Google stosuje się różnorodne mechanizmy zapewniające uznanie, począwszy od standardów obowiązujących w całej firmie, takich jak ocena wydajności pracy i kryteria awansu, a skończywszy na nagrodach wręczanych sobie wzajemnie przez pracowników firmy.

Nasza „drabinka" stosowana w przypadku inżynierii oprogramowania, której używamy do dokładnego ustalania nagród, takich jak rekompensaty i awanse w firmie, zachęca inżynierów do dzielenia się wiedzą, gdyż otwarcie wspomina się o oczekiwaniach powiązanych z nagrodami. Na wyższych szczeblach „drabinki", zajmowanych przez bardzo doświadczonych pracowników, wprost podkreśla się ważność wywierania szerszego wpływu. Związane z tym oczekiwania zwiększają się na coraz wyższych stanowiskach. Na najwyższych szczeblach przykłady przywództwa uwzględniają następujące przypadki:

- Przygotowywanie przyszłych liderów przez bycie mentorami pracowników z mniejszym stażem oraz ułatwianie im rozwoju zarówno z technicznego punktu widzenia, jak i ich roli pełnionej w firmie Google.
- Utrzymywanie i rozwijanie w firmie Google społeczności związanej z oprogramowaniem, w ramach czego dokonuje się oceny kodu i projektów, edukuje i zapewnia rozwój w dziedzinie inżynierii, a także oferuje się innym osobom z branży porady eksperckie.

W rozdziałach 5. i 6. znajdziesz więcej informacji na temat przywództwa.

Oczekiwania związane z poszczególnymi szczeblami „drabinki" stanowisk stanowią metodę odgórnego wprowadzania kultury pracy, ale może być ona też kształtowana w sposób oddolny. Program dodatków dla pracowników tego samego szczebla to w firmie Google jeden ze sposobów zapewniania kultury pracy w wariancie oddolnym. Dodatki dla takich pracowników mają postać nagrody pieniężnej i formalnego uznania, jakie dowolny pracownik firmy Google może wyrazić względem innego pracownika za ponadprzeciętne efekty pracy[11]. Gdy na przykład Ravi przekaże współpracującej z nim Julii dodatek za największe zaangażowanie w utrzymanie listy wysyłkowej (regularnie udzielała odpowiedzi na pytania, z czego skorzystało wiele osób czytających wiadomości), oznacza to, że publicznie uznaje jej działania na rzecz dzielenia się wiedzą i wpływ, jaki to miało poza obrębem jej zespołu. Ponieważ dodatki dla pracowników jednego szczebla są inspirowane przez nich samych, a nie przez zarząd, mogą one powodować istotny i silny efekt oddolny.

Pochwała odgrywa podobną rolę jak dodatki dla pracowników tego samego szczebla: wiąże się z nią publiczne wyrażenie swojej wdzięczności za wkład (zwykle ma to mniejszy wpływ lub wymaga mniejszego nakładu pracy niż działania skutkujące przyznaniem dodatku dla pracowników jednego szczebla), który powoduje zwiększenie stopnia wzajemnego zaangażowania współpracowników.

Gdy jeden pracownik firmy Google wręczy dodatek innemu pracownikowi lub pochwali go, może zdecydować się na przesłanie wiadomości pocztowej dotyczącej nagrody do dodatkowych grup lub wybranych osób, co zwiększy poziom uznania dla efektów pracy współpracownika. Częstą sytuacją jest również skierowanie takiej wiadomości pocztowej przez menedżera skrzynki odbiorczej do członków zespołu, aby mogli świętować wzajemne osiągnięcia.

System, w którym ludzie mogą formalnie i z łatwością wyrażać swoje uznanie dla współpracowników, to narzędzie o dużych możliwościach motywowania ich do dalszego wykonywania wspaniałych rzeczy, jakimi się zajmują. Istotny nie jest dodatek, lecz wdzięczność współpracownika.

[11] Dodatki dla pracowników tego samego szczebla uwzględniają nagrodę w postaci gotówki oraz certyfikat, a także umieszczenie na stałe w rejestrze nagrodzonych pracowników firmy Google znajdującym się w jej narzędziu wewnętrznym o nazwie gThanks.

## Ustanawianie kanonicznych źródeł informacji

Kanoniczne źródła informacji są scentralizowanymi i obowiązującymi w skali całej firmy zbiorami informacji, które zapewniają metodę standaryzowania i propagowania wiedzy eksperckiej. Tego typu źródła sprawdzają się najlepiej w przypadku informacji odnoszących się do wszystkich inżynierów w organizacji. W przeciwnym razie takie źródła mają skłonność do tworzenia „wysp" informacji. Na przykład przewodnik dotyczący konfigurowania podstawowego przepływu informacji procesu projektowania powinien zostać ustanowiony jako kanoniczny. Z kolei przewodnik umożliwiający uruchomienie lokalnej instancji narzędzia Frobber jest bardziej odpowiedni tylko dla inżynierów, którzy z niego korzystają.

Ustanawianie kanonicznych źródeł informacji wymaga poniesienia większych inwestycji niż w przypadku utrzymywania bardziej lokalnych informacji, takich jak dokumentacja zespołu, ale też uzyskuje się większe korzyści. Zapewnienie scentralizowanych odnośników dla całej organizacji sprawia, że łatwiejsze i bardziej przewidywalne staje się wyszukiwanie informacji wymaganych w szerszym zakresie. Ponadto przeciwdziała to problemom związanym z fragmentacją informacji, które mogą się pojawić, gdy wiele zespołów zmagających się z podobnymi trudnościami tworzy własne, często niewspółgrające ze sobą przewodniki.

Ponieważ informacje kanoniczne cechują się wysokim stopniem widoczności, a ponadto mają na celu umożliwienie ogólnego zrozumienia na poziomie organizacji, ważne jest, aby zawartość była aktywnie utrzymywana i sprawdzana przez ekspertów z danej dziedziny. Im bardziej złożone zagadnienie, tym istotniejsze jest, żeby zawartość kanoniczna miała wyraźnie ustalonych właścicieli. Osoby czytające z dobrymi chęciami mogą stwierdzić, że coś jest nieaktualne, ale mogą nie być kompetentne do tego, aby dokonać znaczących zmian strukturalnych niezbędnych do skorygowania tego nawet wtedy, gdy dostępne narzędzia ułatwiają zasugerowanie aktualizacji.

Tworzenie i utrzymywanie scentralizowanych, kanonicznych źródeł informacji jest kosztowne i czasochłonne, a ponadto na poziomie organizacji nie musi być konieczne udostępnianie całej zawartości. Zastanawiając się, jakie nakłady inwestycyjne ponieść w przypadku tego zasobu, weź pod uwagę grupę odbiorców. Kto skorzysta z takich informacji? Ty? Twój zespół? Obszar powiązany z produktem? Wszyscy inżynierowie?

### Przewodniki dla projektantów

Firma Google oferuje inżynierom szeroki i rozbudowany zestaw oficjalnych przewodników, w tym przewodniki dotyczące stylów (*https://google.github.io/styleguide/*), oficjalne zestawienie najlepszych praktyk z zakresu inżynierii oprogramowania[12], wytyczne odnoszące się do inspekcji kodu[13] i testowania[14], a także wskazówki tygodnia TotW (*Tips of the Week*)[15].

[12] Na przykład dostępne w firmie Google książki z dziedziny inżynierii oprogramowania.

[13] Zajrzyj do rozdziału 9.

[14] Zajrzyj do rozdziału 11.

[15] Dostępne dla wielu języków. W przypadku języka C++ dostępne poza firmą, pod adresem *https://abseil.io/tips*.

Zbiór informacji jest tak duży, że niepraktyczne byłoby oczekiwanie, że inżynierowie zaznajomią się z nimi wszystkimi od początku do końca. Ponadto mieliby znacznie mniejsze możliwości przyswojenia na raz tak dużej ilości informacji. Zamiast tego ekspert zaznajomiony już z wytycznymi może wysłać odnośnik zaprzyjaźnionemu inżynierowi, który może następnie przeczytać udostępnione informacje i dowiedzieć się więcej. Ekspert oszczędza czas, gdyż nie musi osobiście objaśniać praktyk obowiązujących w całej firmie, a osoba ucząca się wie już, że istnieje kanoniczne źródło godnych zaufania informacji, z których może skorzystać, gdy tylko zaistnieje taka potrzeba. Taki proces pozwala skalować rozpowszechnianie wiedzy, ponieważ daje ekspertom możliwość identyfikowania i realizowania konkretnego zapotrzebowania na informacje przez korzystanie ze wspólnych i skalowalnych zasobów.

### Odnośniki go/

Odnośniki *go/* (czasami określane mianem odnośników *goto/*) to skrócony zapis wewnętrznych adresów URL firmy Google[16]. Większość wewnętrznych materiałów referencyjnych w firmie zawiera co najmniej jeden wewnętrzny odnośnik *go/*. Na przykład odnośnik *go/spanner* zapewnia informacje o usłudze Spanner, a odnośnik *go/python* kieruje do przewodnika dla projektantów firmy Google używających języka Python. Zawartość może być dostępna w dowolnym repozytorium (g3doc, Dysk Google, Google Sites itp.), ale użycie odnośnika *go/*, który do niej kieruje, zapewnia przewidywalną metodę uzyskiwania dostępu. Wiąże się z tym kilka następujących korzyści:

- Odnośniki *go/* są tak krótkie, że z łatwością można udostępniać je w ramach konwersacji („Powinieneś sprawdzić odnośnik *go/frobber*!"). Jest to znacznie prostsze niż konieczność znajdowania odnośnika, a następnie wysyłania wiadomości do wszystkich zainteresowanych osób. Dysponowanie niepowodującym utrudnień sposobem udostępniania materiałów referencyjnych powoduje, że bardziej prawdopodobne jest to, że przede wszystkim będzie miało miejsce dzielenie się wiedzą.
- Odnośniki *go/* zapewniają trwały odsyłacz do zawartości nawet wtedy, gdy zmieni się bazowy adres URL. Jeśli właściciel zawartości przeniesie ją do innego repozytorium (na przykład z serwisu Google docs na platformę g3doc), może po prostu zaktualizować docelowy adres URL odnośnika *go/*, który sam pozostanie niezmieniony.

Odnośniki *go/* są tak zakorzenione w kulturze pracy w firmie Google, że zaistniał godny uznania cykl: pracownik firmy szukający informacji o narzędziu Frobber prawdopodobnie sprawdzi najpierw odnośnik *go/frobber*. Jeżeli nie skieruje on (jak tego się oczekuje) do przewodnika dla projektantów korzystających z tego narzędzia, pracownik przeważnie będzie sam próbował ustawić właściwy odnośnik. W rezultacie pracownicy firmy Google mogą zwykle przy pierwszej próbie odgadnąć właściwy odnośnik *go/*.

### Codelabs

Serwis Codelabs firmy Google oferuje praktyczne kursy instruktażowe, które umożliwiają inżynierom poznanie nowych zagadnień lub procesów. W kursach połączono objaśnienia, działający przykładowy

[16] Odnośniki *go/* nie mają związku z językiem Go.

kod uwzględniający najlepsze praktyki oraz ćwiczenia związane z pisaniem kodu[17]. Kanoniczny zbiór kursów dotyczących technologii powszechnie używanych w firmie Google jest dostępny za pośrednictwem odnośnika *go/codelab*. Zanim zostaną opublikowane, kursy te podlegają kilku turom formalnej inspekcji i testowania. Kursy z serwisu Codelabs stanowią interesujący element pośredni między statyczną dokumentacją i zajęciami prowadzonymi przez instruktora, a ponadto uwzględniają najlepsze i najgorsze cechy obu rozwiązań. Praktyczna natura kursów sprawia, że są one bardziej angażujące niż tradycyjna dokumentacja, ale inżynierowie w dalszym ciągu korzystają z nich w trybie na żądanie i kończą je we własnym zakresie. Utrzymanie kursów jest jednak kosztowne, a ponadto nie są one dostosowane do konkretnych potrzeb uczącego się.

### Analiza statyczna

Narzędzia do analizy statycznej zapewniają skuteczną metodę dzielenia się najlepszymi praktykami, które mogą być uwzględniane w sposób programowy. Każdy język programowania oferuje własne, specyficzne narzędzia do analizy statycznej, ale mają one to samo ogólne przeznaczenie, czyli informowanie twórców i inspektorów kodu o możliwych sposobach ulepszenia go zgodnie z wytycznymi dotyczącymi stylu i najlepszymi rozwiązaniami. W niektórych narzędziach wykonano krok dalej i zaoferowano zautomatyzowane uwzględnianie tych ulepszeń w kodzie.

W rozdziale 20. zamieszczono szczegóły dotyczące narzędzi do analizy statycznej, a także wyjaśniono, jak są one używane w firmie Google.

Przygotowanie narzędzi do analizy statycznej wymaga wcześniejszego poniesienia inwestycji. Ale gdy tylko będą one dostępne, umożliwią efektywne skalowanie. Po dodaniu do narzędzia opcji uwzględniającej najlepsze rozwiązanie każdy inżynier korzystający z narzędzia staje się świadomy jego dostępności. Zwalnia to również inżynierów z konieczności poznawania innych rzeczy: czas i nakład pracy, które zostałyby przeznaczone na ręczne zaznajomienie się (obecnie odbywa się automatycznie) z najlepszym rozwiązaniem, mogą zostać wykorzystane do nauczenia się czegoś innego. Narzędzia do analizy statycznej umożliwiają poszerzenie wiedzy inżynierów. Pozwalają one organizacji zastosować więcej najlepszych rozwiązań, a ponadto w bardziej spójny sposób byłoby to możliwe w innym przypadku.

## Bycie na bieżąco

Niektóre informacje są kluczowe do realizowania przez kogoś swoich obowiązków (np. znajomość metody tworzenia typowego przepływu informacji procesu projektowania). Inne informacje, takie jak aktualizacje dotyczące popularnych narzędzi produkcyjnych, są mniej kluczowe, lecz nadal przydatne. W przypadku tego typu wiedzy forma nośnika służącego do udostępniania dostarczanych informacji zależy od ich ważności. Na przykład użytkownicy oczekują, że oficjalna dokumentacja będzie cały czas aktualna. Nie mają zwykle jednak takich oczekiwań względem zawartości biuletynu, który zatem może być utrzymywany przez jego właściciela przy mniejszym nakładzie pracy.

[17] Zewnętrzna wersja serwisu Codelabs jest dostępna pod adresem *https://codelabs.developers.google.com/*.

### Biuletyny

W skali całej firmy Google funkcjonuje wiele biuletynów wysyłanych do wszystkich inżynierów. Są wśród nich takie jak *EngNews* (wiadomości dla inżynierów), *Ownd* (wiadomości dotyczące kwestii prywatności i bezpieczeństwa) oraz *Google's Greatest Hits* (raport uwzględniający najbardziej interesujące awarie danego kwartału). Jest to dobry sposób przekazywania informacji, które interesują inżynierów, ale nie są kluczowe. W przypadku tego typu aktualizacji stwierdziliśmy, że biuletyny powodują większe zaangażowanie, gdy są wysyłane rzadziej, a ponadto zawierają bardziej przydatną i interesującą treść. W przeciwnym razie biuletyny mogą być postrzegane jako spam.

Nawet pomimo tego, że większość biuletynów w firmie Google jest wysyłana pocztą elektroniczną, niektóre z nich są dystrybuowane w bardziej kreatywny sposób. *Testing on the Toilet* (wskazówki dotyczące testowania) i *Learning on the Loo* (wskazówki związane z wydajnością) to jednostronicowe biuletyny umieszczane w kabinach toalet. Ta unikatowa forma przekazywania informacji ułatwia odróżnienie obu tych biuletynów od wszystkich pozostałych. Ponadto wszystkie wydania tych dwóch biuletynów są archiwizowane w trybie online.

W rozdziale 11. przedstawiono historię tego, jak powstał biuletyn *Testing on the Toilet*.

### Społeczności

Aby dzielić się wiedzą, pracownicy firmy Google lubią tworzyć z myślą o różnych zagadnieniach społeczności obejmujące swoim zasięgiem różne działy firmy. Takie otwarte kanały ułatwiają uczenie się od osób spoza bezpośredniego otoczenia oraz unikanie „wysp" informacji i duplikowania. Szczególnie popularne są grupy firmy Google: istnieją tysiące grup wewnętrznych o różnych stopniach formalności. Niektóre grupy zajmują się rozwiązywaniem problemów. Inne grupy, takie jak Code Health, są przeznaczone bardziej do dyskusji i porad. Wewnętrzna wersja usługi Google+ również cieszy się powodzeniem wśród pracowników firmy Google jako źródło nieformalnych informacji, ponieważ ludzie będą publikować interesujące analizy techniczne lub szczegóły projektów, którymi się zajmują.

## Czytelność — standaryzowane doradztwo w ramach inspekcji kodu

W firmie Google termin „czytelność" odnosi się do czegoś więcej niż tylko do czytelności kodu. Tym mianem określa się standaryzowany i obowiązujący w całej firmie proces doradztwa mający na celu szerzenie najlepszych praktyk dotyczących języków programowania. Czytelność uwzględnia szeroki zakres kompetencji, który obejmuje między innymi idiomy językowe, strukturę kodu, projekt interfejsów API, właściwe użycie wspólnych bibliotek, dokumentację i pokrycie kodu.

Na początku z czytelnością był związany wysiłek jednej osoby. W początkach istnienia firmy Google Craig Silverstein (pracownik z identyfikatorem 3) spotykał się osobiście z każdym nowym pracownikiem i wiersz po wierszu dokonywał „oceny czytelności" jego pierwszej poważnej zmiany w kodzie,

która miała zostać wprowadzona. Była to bardzo drobiazgowa ocena obejmująca wszystko, począwszy od możliwych sposobów ulepszenia kodu, a skończywszy na konwencjach dotyczących białych znaków. Choć sprawiło to, że baza kodu firmy Google zyskała spójną postać, ważniejsze było, że umożliwiło to poznawanie najlepszych praktyk, zwracanie uwagi na to, jaka była dostępna wspólna infrastruktura, a ponadto dzięki temu nowi pracownicy dowiadywali się, na czym polega tworzenie kodu w firmie Google.

Nieuchronne było to, że poziom zatrudnienia w firmie Google zwiększy się ponad możliwości poradzenia sobie z tym przez jedną osobę. Tak wielu inżynierów uznało ten program za wartościowy, że sami zgłaszali się, aby poświęcić swój czas na dalsze rozwijanie go. Obecnie około 20% inżynierów w firmie uczestniczy w dowolnym momencie w procesie czytelności, będąc albo inspektorami, albo twórcami kodu.

## Czym jest proces oceny czytelności?

W firmie Google inspekcja kodu jest obowiązkowa. Każda lista zmian wymaga *potwierdzenia czytelności*[18], co oznacza, że zaakceptował ją ktoś dysponujący *certyfikatem czytelności* dla danego języka. Twórcy posiadający certyfikat niejawnie zapewniają potwierdzenie czytelności własnych list zmian. W innej sytuacji co najmniej jeden kwalifikowany inspektor musi jawnie utworzyć potwierdzenie czytelności danej listy zmian. Wymóg ten został dodany po tym, jak firma Google powiększyła się do tego stopnia, że nie było już dłużej możliwe wymuszanie, aby każdy inżynier otrzymał wyniki inspekcji kodu, które uwzględniały najlepsze praktyki na żądanym poziomie.

W rozdziale 9. dokonano przeglądu procesu inspekcji kodu w firmie Google, a także wyjaśniono, co w tym kontekście oznacza termin „potwierdzenie”.

W firmie Google dysponowanie certyfikatem czytelności jest przeważnie rozumiane jako spełnienie wymogu czytelności z punktu widzenia danego języka. Inżynierowie mający taki certyfikat potwierdzili, że cały czas tworzą przejrzysty, idiomatyczny i możliwy do utrzymania kod, który wykorzystuje najlepsze praktyki firmy Google oraz styl pisania kodu w danym języku. Osoby te osiągają to dzięki poddawaniu list zmian procesowi oceny czytelności, podczas którego centralna grupa *inspektorów zajmujących się czytelnością* ocenia listy zmian i przekazuje swoje opinie o tym, w jakim stopniu demonstrują one różne obszary biegłości. Gdy twórcy zmian uwzględniają wytyczne dotyczące czytelności, otrzymują coraz mniej komentarzy odnoszących się do ich list zmian, a ostatecznie umożliwiają zakończenie procesu i formalne uzyskanie potwierdzenia czytelności. Z czytelnością wiąże się zwiększona odpowiedzialność: inżynierowie z potwierdzeniem czytelności są obdarzeni zaufaniem pozwalającym im na dalsze wykorzystywanie swojej wiedzy w tworzonym przez siebie kodzie, a ponadto mogą oni być inspektorami kodu napisanego przez innych inżynierów.

W firmie Google około 1 – 2% inżynierów to inspektorzy oceniający czytelność. Wszyscy inspektorzy są wolontariuszami, a każda osoba z certyfikatem czytelności może oczywiście sama wyznaczyć się

[18] *Lista zmian* zawiera pliki tworzące zmianę dokonywaną w systemie kontroli wersji. Lista zmian to synonim *zestawu zmian* (*https://pl.wikipedia.org/wiki/Zestaw_zmian*).

do roli inspektora zajmującego się kwestią czytelności. Tacy inspektorzy muszą utrzymywać najwyższe standardy, gdyż oczekuje się od nich nie tylko dużych kompetencji językowych, ale też zdolności do uczenia się w trakcie przeprowadzania inspekcji kodu. Oczekuje się, że takie osoby będą traktować kwestię oceny czytelności jako przede wszystkim proces doradzania i współpracy, a nie kontroli lub wywoływania antagonizmów. Inspektorzy od kwestii czytelności oraz twórcy listy zmian są zachęcani do prowadzenia dyskusji w trakcie trwania procesu inspekcji. Inspektorzy zapewniają odpowiednie cytaty powiązane z ich komentarzami, dzięki czemu twórcy mogą poznać powody, dla których skorzystano z wytycznych dotyczących stylów („płot Chestersona"). Jeśli powód zastosowania jakiejkolwiek wytycznej jest niejasny, twórcy powinni poprosić o wyjaśnienie („zadawaj pytania").

Ocena czytelności to proces świadomie realizowany przez ludzi i mający na celu skalowanie wiedzy w standaryzowany sposób, który jednocześnie jest personalizowany. Będąc połączeniem uzupełniającej się wiedzy w formie pisemnej i wiedzy w obrębie grupy, czytelność oferuje zalety dokumentacji w postaci pisemnej, do której można uzyskać dostęp za pomocą odwołań umożliwiających cytowanie, a także korzyści zapewniane przez doświadczonych inspektorów wiedzących, jakie wytyczne mają zostać przytoczone. Wytyczne kanoniczne i zalecenia językowe są obszernie udokumentowane (co jest czymś dobrym!), ale zbiory informacji są tak duże[19], że mogą okazać się przytłaczające, zwłaszcza dla początkujących.

## Dlaczego korzysta się z tego procesu?

Kod jest czytany znacznie częściej, niż jest tworzony. Kwestia ta znacznie się nasila, gdy pod uwagę weźmie się skalę firmy Google i nasze (bardzo duże) repozytorium monolityczne[20]. Każdy inżynier może sprawdzać zasoby wiedzy, jaką jest kod napisany przez członków innych zespołów, a także dokształcić się dzięki niej. Zaawansowane narzędzia, takie jak Kythe (*https://kythe.io/*), ułatwiają znajdowanie informacji referencyjnych w całej bazie kodu (zajrzyj do rozdziału 17.). Ważną cechą udokumentowanych, najlepszych praktyk (zajrzyj do rozdziału 8.) jest to, że zapewniają one spójne standardy, na których może bazować cały kod tworzony w firmie Google. W przypadku tych standardów czytelność pełni rolę mechanizmu zarówno rozpowszechniania, jak i wymuszania.

Jedną z podstawowych korzyści programu dotyczącego czytelności jest to, że udostępnia on inżynierom więcej niż tylko wiedzę obecną w obrębie ich własnego zespołu. Aby potwierdzić czytelność dla danego języka, inżynierowie muszą przesłać listy zmian do centralnej grupy inspektorów zajmujących się kwestią czytelności, którzy oceniają kod w skali całej firmy. Centralizowanie tego procesu powoduje wystąpienie znaczącego kompromisu: przy powiększaniu się organizacji program jest ograniczony bardziej do skalowania liniowego niż podliniowego, ale ułatwia to wymuszanie spójności oraz unikanie „wysp" informacji i eliminowanie (często niezamierzonego) odchodzenia od ustalonych norm.

[19] W roku 2019 sam przewodnik firmy Google dotyczący stylów języka C++ liczył 40 stron. Dodatkowy materiał tworzący kompletny zbiór najlepszych praktyk jest wielokrotnie bardziej obszerny.

[20] Aby dowiedzieć się, dlaczego firma Google korzysta z repozytorium monolitycznego, zajrzyj na stronę dostępną pod adresem *https://cacm.acm.org/magazines/2016/7/204032-why-google-storesbillions-of-lines-of-code-in-a-single-repository/fulltext*. Zauważ też, że kod stworzony przez firmę nie jest w całości przechowywany w tym repozytorium. Opisana tutaj kwestia czytelności ma zastosowanie tylko względem repozytorium monolitycznego, ponieważ jest to pojęcie dotyczące spójności w obrębie repozytorium.

Nie sposób przecenić wartości w postaci spójności całej bazy kodu: nawet w przypadku dziesiątków tysięcy inżynierów tworzących kod przez dekady wartość taka zapewnia, że kod napisany w danym języku będzie wyglądać podobnie w obrębie całego zbioru. Pozwala to osobom czytającym kod skoncentrować się na tym, co on powoduje, a nie rozpraszać się rozmyślaniem, dlaczego wygląda on inaczej niż kod, do którego one przywykły. Twórcy zmian na dużą skalę (zajrzyj do rozdziału 22.) mogą łatwiej wprowadzać zmiany w całym repozytorium monolitycznym, które obejmuje wyniki pracy tysięcy zespołów. Ludzie mogą zmieniać zespoły i być pewnym tego, że sposób, w jaki członkowie nowego zespołu korzystają z danego języka, nie będzie różnić się znacząco od sposobu wykorzystywanego w ich poprzednim zespole.

Z tymi korzyściami związane są pewne koszty: w porównaniu z innymi formami przekazu informacji, takimi jak dokumentacja i zajęcia, ocena czytelności to intensywny proces, ponieważ jest obowiązkowy i wymuszany przez narzędzia firmy Google (zajrzyj do rozdziału 19.). Koszty te nie są trywialne i obejmują następujące rzeczy:

- Nasilone tarcia w zespołach, w których żaden ich członek nie ma możliwości oceny czytelności. Wynika to z tego, że w takiej sytuacji zespoły te są zmuszone do poszukania inspektorów na zewnątrz, aby uzyskać potwierdzenie czytelności dla swoich list zmian.
- Możliwość wystąpienia dodatkowych rund procesu inspekcji kodu w przypadku twórców, którzy wymagają oceny kwestii czytelności.
- Ograniczone możliwości skalowania wynikające z procesu realizowanego przez ludzi. W tej sytuacji przy rozwoju organizacji możliwe jest tylko skalowanie liniowe, ponieważ proces jest zależny od inspektorów dokonujących inspekcji kodu wymagających specjalistycznej wiedzy.

W takim razie można się zastanawiać, czy korzyści przeważają nad kosztami. Pojawia się też czynnik w postaci czasu: pełny efekt zestawienia ze sobą korzyści i kosztów nie bazuje na tej samej skali czasowej. W ramach prezentowanego programu tworzy się zamierzony kompromis polegający na tym, że w zamian za zwiększenie krótkoterminowo opóźnienia związanego z inspekcją kodu i kosztów początkowych długofalowo uzyskuje się wyższej jakości kod, spójność kodu w skali całego repozytorium oraz większe kompetencje inżynierów. Szerszej skali czasowej korzyści towarzyszy oczekiwanie, że kod jest tworzony z myślą o czasie jego istnienia wynoszącym potencjalnie wiele lat, jeśli nie dekad[21].

Jak w przypadku większości lub być może wszystkich procesów inżynieryjnych, zawsze jest miejsce na ulepszenia. Niektóre koszty mogą zostać zmniejszone za pomocą narzędzi. Wiele komentarzy dotyczących czytelności pozwala rozwiązać problemy, które mogły zostać wykryte w wariancie analizy statycznej i opatrzone automatycznie komentarzem przez narzędzia do tego typu analizy. Wraz z dalszym inwestowaniem w analizę statyczną inspektorzy oceniający kwestię czytelności mogą w coraz większym stopniu koncentrować się na zagadnieniach wyższego poziomu, takich jak to, czy konkretny blok kodu będzie zrozumiały dla czytających go osób spoza firmy, które nie są dobrze zaznajomione z bazą kodu, a nie zajmować się rzeczami możliwymi do wykrycia w sposób automatyczny (np. stwierdzanie, czy wiersz kodu zawiera na końcu biały znak).

[21] Z tego powodu kod, w przypadku którego wiadomo, że ma krótki czas istnienia, nie jest objęty wymogami czytelności. Odpowiednie przykłady uwzględniają katalog *experimental/* (przewidziany specjalnie dla kodu eksperymentalnego, który nie może być umieszczany w środowisku produkcyjnym) oraz program Area 120 (*https://area120.google.com/*), czyli miejsce służące do rozwijania eksperymentalnych produktów firmy Google.

Aspiracje nie są jednak wystarczające. Ocena czytelności to kontrowersyjny program: niektórzy inżynierowie narzekają, że jest to niepotrzebna przeszkoda biurokratyczna i kiepski sposób na wykorzystanie ich czasu. Czy kompromisy towarzyszące czytelności są tego warte? Aby uzyskać odpowiedź, skontaktowaliśmy się z członkami zespołu EPR (*Engineering Productivity Research*), który zajmuje się badaniami wydajności pracy inżynierów.

Zespół ten przeanalizował dogłębnie kwestię czytelności, uwzględniając między innymi to, czy proces utrudniał ludziom pracę, czy nauczyli się czegoś albo czy zmienili sposób postępowania po jego zakończeniu. Przeprowadzone badania pokazały, że czytelność ma ogólnie pozytywny wpływ na szybkość pracy inżynierów. Listy zmian sporządzone przez twórców mających certyfikat czytelności statystycznie wymagają poświęcenia znacznie mniejszej ilości czasu na inspekcję i wprowadzanie niż listy zmian utworzone przez osoby niedysponujące takim certyfikatem[22]. Zadowolenie wynikające z jakości kodu zgłaszane przez samych inżynierów, gdy nie ma dostępnych bardziej obiektywnych metod pomiaru jakości kodu, jest większe wśród inżynierów z certyfikatem czytelności niż u tych, którzy go nie posiadają. Znacząca większość inżynierów, którzy w całości realizują program, zgłasza zadowolenie z procesu i uważa go za wart zachodu. Osoby te informują, że dzięki inspektorom uczą się i zmieniają sposób swojego postępowania, aby uniknąć problemów z czytelnością w trakcie tworzenia *i* oceniania kodu.

W rozdziale 7. dogłębnie omówiono powyższe analizy oraz wewnętrzne badania firmy Google dotyczące wydajności pracy inżynierów.

W firmie Google istnieje bardzo silna kultura związana z procesem inspekcji kodu, a kwestia czytelności stanowi jej naturalne rozszerzenie. Czytelność rozwinęła się z pasji jednego inżyniera do postaci formalnego programu, w ramach którego eksperci pełnią rolę mentorów dla wszystkich inżynierów firmy Google. Kwestia czytelności rozwijała się oraz zmieniała wraz z powiększaniem się firmy i nadal będzie rozwijana zależnie od jej zmieniających się potrzeb.

# Podsumowanie

Pod pewnymi względami wiedza to najważniejszy (choć nieuchwytny) kapitał organizacji z branży inżynierii oprogramowania. Dzielenie się tą wiedzą jest kluczowe do tego, aby w obliczu zmiany organizacja okazała się odporna i redundantna. Kultura promująca dzielenie się wiedzą w otwarty i szczery sposób umożliwia skuteczne rozpowszechnianie jej w całej organizacji, a ponadto sprawia, że z upływem czasu może być ona skalowana. W większości przypadków inwestycje mające na celu łatwiejsze udostępnianie wiedzy pozwalają w czasie istnienia firmy uzyskać rozmaite korzyści.

[22] Obejmuje to kontrolowanie pod kątem różnych czynników, w tym stałego etatu w firmie Google, a także tego, że w porównaniu z listami zmian autorstwa osób dysponujących już certyfikatem czytelności takie listy utworzone przez twórców niemających tego certyfikatu wymagają zwykle dodatkowych rund procesu inspekcji.

# W skrócie

- *Bezpieczeństwo psychologiczne* to fundament procesu budowania środowiska umożliwiającego dzielenie się wiedzą.
- Zacznij od czegoś małego: zadawaj pytania i notuj różne rzeczy.
- Ułatwiaj ludziom uzyskanie pomocy, jakiej potrzebują, zarówno od ekspertów, jak i dzięki udokumentowanym informacjom referencyjnym.
- Na poziomie systemowym zachęcaj i nagradzaj osoby, które poświęcają swój czas na uczenie innych i dzielenie się swoją wiedzą poza obrębem swojego zespołu lub organizacji.
- Nie ma złotego środka: kształtowanie kultury dzielenia się wiedzą wymaga połączenia wielu strategii, a konkretna kombinacja, która sprawdzi się najlepiej w przypadku Twojej firmy, będzie się prawdopodobnie zmieniać z upływem czasu.

# Piszesz kod? To ważne zadanie — bierz przykład z najlepszych!

Inżynieria oprogramowania jest pojęciem znacznie szerszym od kodowania: oznacza wszystkie niezbędne narzędzia i procesy stosowane przez organizację do tworzenia oprogramowania. To daje możliwość zachowania wartości kodu w dłuższej perspektywie czasu i pozwala ustanowić bardziej rygorystyczne zasady tworzenia oprogramowania, a dzięki temu sam kod jest podatniejszy na zmiany. Innymi słowy, inżynieria oprogramowania polega na optymalnym integrowaniu i organizowaniu tworzenia aplikacji — od koncepcji, poprzez tworzenie, wdrażanie i utrzymywanie, po jej wycofywanie.

To nie jest podręcznik dla programistów. Celem autorów jest zaprezentowanie jedynej w swoim rodzaju perspektywy firmy Google, od lat rozwijającej trwały ekosystem oprogramowania, co pozwoliło zebrać pożyteczne wnioski dotyczące skali działalności i czasu jej trwania. W książce zwrócono uwagę na to, że proces tworzenia oprogramowania jest wysiłkiem zespołowym, omówiono najlepsze praktyki związane z utrzymywaniem bazy kodu o dużych rozmiarach i długim stażu, pokazano także narzędzia, które mogą się okazać przydatne w jej utrzymywaniu. Omówione tu zagadnienia uwzględniają doświadczenia, jakie typowy inżynier oprogramowania zdobywa w ramach swojej pracy, służą też wskazaniu różnorodnych sposobów rozwiązywania poszczególnych problemów.

## Najciekawsze zagadnienia:

- unikatowa kultura pracy w Google
- procesy i narzędzia stosowane w Google
- metody zwiększania odporności kodu na upływ czasu
- wpływ skali oprogramowania na organizację pracy inżynierów
- kompromisy w procesie podejmowania decyzji projektowych

**Titus Winters** jest starszym inżynierem oprogramowania w Google. Kieruje pracami zespołu odpowiedzialnego za bazę kodu C++ firmy Google.

**Tom Manshreck** jest członkiem zespołu zajmującego się bibliotekami języka C++ w Google. Odpowiada za tworzenie dokumentacji technicznej.

**Hyrum Wright** jest inżynierem oprogramowania w Google. Kieruje grupą, która tworzy narzędzia do zautomatyzowanego wprowadzania zmian.

Helion
helion.pl
HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl
KOD KORZYŚCI
*Sięgnij po więcej!*
ISBN 978-83-283-9971-6
9 788328 399716
Cena: 129,00 zł