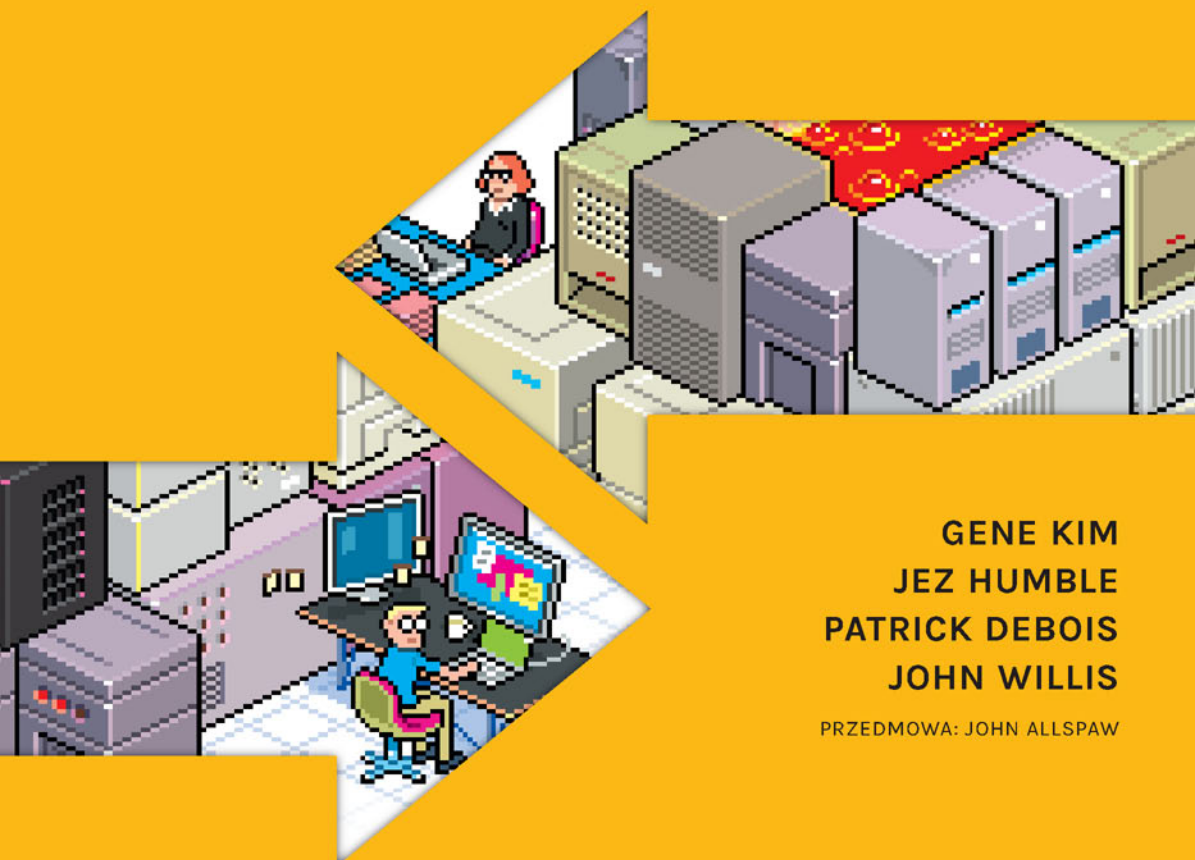


# DevOps

ŚWIATOWEJ KLASY  
ZWINNOŚĆ, NIEZAWODNOŚĆ  
I BEZPIECZEŃSTWO  
W TWOJEJ ORGANIZACJI



GENE KIM  
JEZ HUMBLE  
PATRICK DEBOIS  
JOHN WILLIS

PRZEDMOWA: JOHN ALLSPAW

onepress

Helion 

Tytuł oryginału: The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations

Tłumaczenie: Radosław Meryk

ISBN: 978-83-283-3453-3

Copyright © 2016 by Gene Kim, Jez Humble, Patrick Debois, and John Willis  
All rights reserved

Polish edition copyright © 2017 by Helion SA  
All rights reserved.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Autor studium przypadku zespołu 18F na stronie 355 przekazał swoje dzieło do domeny publicznej zrzekając się praw autorskich oraz praw zależnych do dzieła na terenie całego świata, w zakresie dopuszczonym przez prawo. Można kopiować, modyfikować oraz rozpowszechniać i wykonywać studium przypadku zespołu 18F, także do celów komercyjnych, bez występowania o zgodę.

Wydawnictwo HELION  
ul. Kościuszki 1c, 44-100 GLIWICE  
tel. 32 231 22 19, 32 230 98 63  
e-mail: [helion@helion.pl](mailto:helion@helion.pl)  
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie/devops>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

# SPIS TREŚCI

|  |           |
|--|-----------|
| Przedmowa. Aha! .....  | 7         |
| Słowo wstępne .....  | 15        |
| Wyobraź sobie świat, w którym Dev i Ops tworzą DevOps.<br>Wprowadzenie do podręcznika DevOps .....                             | 17        |
| <b>CZĘŚĆ I. TRZY DROGI .....</b>   | <b>33</b> |
| 1. Agile, ciągle dostarczanie i trzy drogi .....   | 39        |
| 2. Pierwsza droga: Zasady przepływu.....   | 47        |
| 3. Druga droga: Zasady sprzężenia zwrotnego.....   | 59        |
| 4. Trzecia droga: Zasady ciągłego uczenia się i eksperymentowania.....   | 69        |
| <b>CZĘŚĆ II. OD CZEGO ZACZAĆ? .....</b>  | <b>79</b> |
| 5. Wybór strumieni wartości, od których należy zacząć.....   | 83        |
| 6. Zrozumienie pracy w strumieniu wartości, zapewnienie jej widoczności<br>i rozszerzenie zrozumienia na całą organizację..... | 93        |
| 7. Projektowanie organizacji i jej architektury<br>z uwzględnieniem praw Conwaya.....  | 107       |
| 8. Jak uzyskać świetne efekty poprzez zintegrowanie zadań działu Ops<br>z codzienną pracą działu Dev? .....                    | 125       |

## CZEŚĆ III. PIERWSZA DROGA

|   |     |
|---|-----|
| <i>TECHNICZNE PRAKTYKI PRZEPŁYWU</i> .....                  | 137 |
| 9. Podstawy potoku wdrożeń .....                            | 141 |
| 10. Szybkie i niezawodne testowanie automatyczne .....      | 153 |
| 11. Wdrożenie i stosowanie praktyk ciągłej integracji ..... | 173 |
| 12. Automatyzacja i zapewnienie wydań niskiego ryzyka ..... | 183 |
| 13. Architektura dla wydań niskiego ryzyka .....            | 209 |

## CZEŚĆ IV. DRUGA DROGA

|  |     |
|--|-----|
| <i>TECHNICZNE PRAKTYKI SPRZĘŻEŃ ZWROTNYCH</i> .....  | 221 |
| 14. Tworzenie telemetrii umożliwiające dostrzeżenie i rozwiązywanie problemów .....                                | 225 |
| 15. Analizowanie telemetrii w celu lepszego przewidywania problemów i realizowania zadań .....                     | 245 |
| 16. Sprzężenia zwrotne poprawiają bezpieczeństwo wdrażania kodu przez zespoły Dev i Ops .....                      | 259 |
| 17. Integracja technik wytwarzania oprogramowania sterowanego hipotezami i testowania A/B w codziennej pracy ..... | 273 |
| 18. Tworzenie procesów przeglądu i koordynacji w celu poprawy jakości bieżącej pracy .....                         | 281 |

## CZEŚĆ V. TRZECIA DROGA

|  |     |
|--|-----|
| <i>TECHNICZNE PRAKTYKI CIĄGŁEGO UCZENIA SIĘ I EKSPERYMENTOWANIA</i> .....                  | 299 |
| 19. Stworzenie warunków do uczenia się podczas codziennej pracy .....                      | 303 |
| 20. Konwersja lokalnych odkryć w globalne usprawnienia .....                               | 317 |
| 21. Zarezerwuj czas na stworzenie organizacyjnego systemu uczenia się i doskonalenia ..... | 329 |

## CZEŚĆ VI. ZARZĄDZANIE ZMIANAMI

|  |     |
|--|-----|
| <i>I ZAPEWNIENIE ZGODNOŚCI Z PRZEPISAMI</i> .....                        | 339 |
| 22. Bezpieczeństwo informacji jako codzienne zadanie każdego z nas ..... | 343 |
| 23. Ochrona potoku wdrożeń .....   | 363 |
| Wezwanie do działania. Podsumowanie podręcznika DevOps .....             | 377 |

|                                  |            |
|----------------------------------|------------|
| <b>MATERIAŁY DODATKOWE .....</b> | <b>381</b> |
| Dodatki .....                    | 383        |
| Zasoby dodatkowe.....            | 397        |
| Przypisy końcowe .....           | 401        |
| Skorowidz.....                   | 437        |
| Podziękowania.....               | 445        |
| Biogramy autorów.....            | 449        |



## *Agile, ciągłe dostarczanie i trzy drogi*

W tym rozdziale zaprezentowano wstęp do teorii produkcji Lean, a także **trzy drogi** — zasady, na podstawie których można wywnioskować wszystkie obserwowane zachowania DevOps.

Skoncentrujemy się tutaj przede wszystkim na teorii i zasadach opisujących wiele dziesięcioleci doświadczeń firm produkcyjnych, organizacji wysokiej niezawodności, modeli zarządzania bazujących na wysokim zaufaniu i innych czynników i nurtów, z których wywodzą się praktyki DevOps. W pozostałych rozdziałach książki przedstawiono wynikowe konkretne zasady i wzorce oraz ich praktyczne zastosowanie do strumienia wartości technologii.

### STRUMIEŃ WARTOŚCI PRODUKCJI

Jedną z podstawowych koncepcji Lean jest strumień wartości. Zdefiniujemy go najpierw w kontekście produkcji, a następnie dokonamy jego ekstrapolacji w celu zastosowania go do DevOps oraz strumienia wartości technologii.

Karen Martin i Mike Osterling w swojej książce *Value Stream Mapping: How to Visualize Work and Align Leadership for Organizational Transformation* zdefiniowali strumień wartości jako „sekwencję działań podejmowanych przez organizację w celu realizacji zlecenia klienta” lub „sekwencję działań wymaganych do zaprojektowania, wyprodukowania i dostarczenia towaru lub usługi do klienta z uwzględnieniem przepływów informacji i materiałów”.

W działalności produkcyjnej strumień wartości często jest łatwy do zaobserwowania: zaczyna się po otrzymaniu zamówienia klienta i dostarczeniu surowców do hali produkcyjnej. Aby umożliwić szybkie i przewidywalne terminy realizacji w każdym strumieniu wartości, zwykle kładzie się ciągły nacisk na tworzenie bezproblemowego i równomiernego przepływu pracy przy użyciu takich technik, jak niewielkie partie materiałów, zmniejszenie produkcji niezakończonych (ang. *Work In Process* — **WIP**), przeciwdziałanie przeróbkom, aby mieć pewność, że nie przekazujemy defektów do centrów pracy w dole strumienia, oraz ciągle optymalizowanie systemu pod kątem osiągnięcia globalnych celów.

## STRUMIEŃ WARTOŚCI TECHNOLOGII

Te same zasady i wzorce, które umożliwiły szybki przepływ pracy w procesach fizycznych, odnoszą się również do prac nad technologią (oraz do wszystkich rodzajów pracy umysłowej). W infrastrukturze DevOps strumień wartości technologii zwykle określamy jako proces niezbędny do przekształcenia hipotezy biznesowej na korzystającą z technologii usługę, która dostarcza wartości dla klienta.

Dane wejściowe dla procesu to sformułowanie celu biznesowego, koncepcji, pomysłu lub hipotezy. Proces zaczyna się w chwili, gdy zaakceptujemy prace w rozwoju poprzez dodanie ich do zbioru zadań do wykonania (ang. *backlog*).

Od tego momentu zespoły deweloperskie stosujące typowe podejście Agile lub proces iteracyjny najczęściej przekształcają tę koncepcję na historyjki użytkowników oraz jakąś postać specyfikacji funkcji, które następnie są implementowane w kodzie aplikacji albo tworzonej usłudze. Następnie kod jest przekazywany do repozytorium kontroli wersji, gdzie każda zmiana jest testowana i integrowana z pozostałą częścią systemu oprogramowania.

Ponieważ wartość jest tworzona tylko wtedy, kiedy usługi są uruchamiane w produkcji, to musimy zadbać nie tylko o szybki przepływ, ale także o to, aby wdrożenie nie spowodowało chaosu i zakłóceń, takich jak przerwy w dostawie usług, słaba jakość usług czy też błędy w zabezpieczeniach lub zgodności z przepisami.

## KONCENTRACJA NA CZASIE REALIZACJI WDRAŻANIA

W pozostałej części książki skoncentrujemy uwagę na czasie realizacji wdrożenia — podzbiórze strumienia wartości opisanego powyżej. Ten strumień wartości zaczyna się w chwili, gdy dowolny inżynier\* w strumieniu wartości (który obejmuje dział rozwoju, walidacji, operacji IT i bezpieczeństwa informacji) prześle zmianę do repozyto-

---

\* Idąc dalej, określenie „inżynier” odnosi się do wszystkich inżynierów pracujących w strumieniu wartości, nie tylko deweloperów.



rium kontroli wersji, a kończy, gdy ta zmiana zostanie pomyślnie zastosowana w produkcji, zapewni wartość dla klienta oraz wygeneruje przydatne sprzężenie zwrotne i dane telemetryczne.

Pierwszy etap prac, który obejmuje projektowanie i rozwój, jest zbliżony do fazy rozwoju produktu w Lean (ang. *Lean Product Development*). Jest bardzo zróżnicowany i wysoce niepewny, często wymaga wysokiego stopnia kreatywności i wykonania pracy, która może nigdy nie zostać wykorzystana ponownie. W związku z tym czasy przetwarzania są na tym etapie bardzo zmienne. Dla odróżnienia drugiego etapu prac, który obejmuje testowanie i uruchamianie, jest zbliżony do produkcji Lean (ang. *Lean Manufacturing*). Wymaga on kreatywności i wiedzy oraz dążenia do tego, aby był przewidywalny i mechanistyczny. Celem jest uzyskanie efektów pracy przy jak najmniejszej zmienności (np. krótkie i przewidywalne czasy realizacji, prawie zerowa liczba defektów).

Zamiast dużych partii pracy przetwarzanych sekwencyjnie przez strumień wartości projektowania i rozwoju, a następnie strumień wartości testowania i uruchamiania (tak jak w dużym projekcie realizowanym według metodologii kaskadowej lub podczas rozwijania długowiecznych gałęzi funkcji) naszym celem jest realizowanie fazy testowania i uruchamiania w tym samym czasie, w którym są realizowane projektowanie i rozwój. Takie działania umożliwiają szybki przepływ i zapewniają wysoką jakość. Metoda kończy się sukcesem, gdy pracujemy niewielkimi partiami, budując jakość we wszystkich częściach strumienia wartości\*.

## DEFINIOWANIE CZASU REALIZACJI A CZAS PRZETWARZANIA

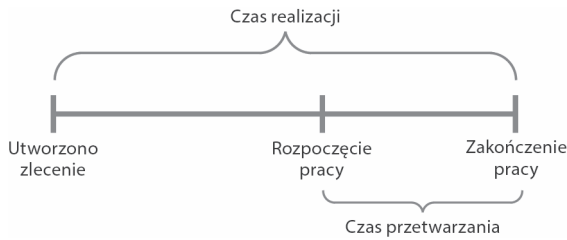
W społeczności Lean czas realizacji (ang. *lead time*) jest jedną z dwóch metryk powszechnie stosowanych do pomiaru wydajności w strumieniu wartości. Drugim jest czas przetwarzania — ang. *processing time* (czasami znany jako *czas zadania* — ang. *task time*)<sup>†</sup>.

O ile czas realizacji zaczyna się w momencie złożenia zamówienia, a kończy się, gdy zamówienie zostanie zrealizowane, o tyle czas przetwarzania rozpoczyna się dopiero wtedy, gdy zaczynamy pracę nad zleceniem klienta — w szczególności pomijany jest czas, gdy zlecenie oczekuje w kolejce na przetworzenie (rysunek 2).

---

\* W rzeczywistości w przypadku stosowania takich technik jak TDD testowanie jest wykonywane przed napisaniem jakiegokolwiek linijki kodu.

<sup>†</sup> Będziemy w tej książce faworyzowali termin „czas przetwarzania” z tych samych powodów, jakie wymienili Karen Martin i Mike Osterling: „Aby zminimalizować zamieszanie, będziemy unikali używania terminu *czas cyklu*, ponieważ istnieje dla niego kilka definicji równoznacznych, np. *czas przetwarzania* lub *częstotliwość wyników*”.

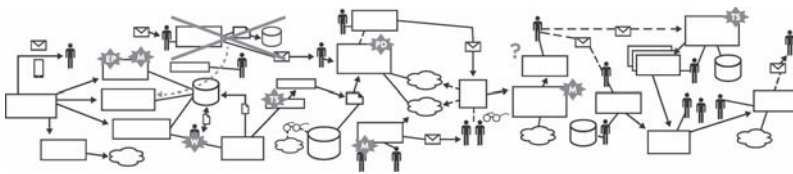


**Rysunek 2.** Czas realizacji a czas przetwarzania w operacji wdrażania

Ponieważ klient odczuwa czas realizacji, to zwykle na nim, a nie na czasie przetwarzania koncentrujemy wysiłek zmierzający do usprawniania procesu. Jednak proporcje pomiędzy czasem przetwarzania a czasem realizacji odgrywają rolę ważnego wskaźnika wydajności — uzyskanie szybkiego przepływu w krótkim czasie prawie zawsze wymaga skrócenia czasu oczekiwania pracy w kolejce.

### TYPOWY SCENARIUSZ. CZASY REALIZACJI WDRAŻANIA WYMAGAJĄCE MIESIĘCY

W biznesie czasami występują sytuacje, gdy czasy realizacji wdrożeń wymagają miesięcy. Jest to szczególnie powszechne w dużych, złożonych organizacjach, pracujących ze ściśle powiązаныmi, monolitycznymi aplikacjami, często z ograniczonymi środowiskami testów integracyjnych, długimi czasami wykonania testów i wdrożeń w środowisku produkcyjnym, wysokim stopniem uzależnienia od ręcznego testowania oraz wielu wymaganych procesów zatwierdzania. W takim przypadku strumień wartości może wyglądać tak, jak pokazano na rysunku 3:



**Rysunek 3.** Strumień wartości technologii z czasem wdrażania wynoszącym trzy miesiące (źródło: Damon Edwards, „DevOps Kaizen”, 2015)

W przypadku długich czasów realizacji wdrożeń na prawie każdym etapie strumienia wartości wymagana jest „heroiczna” praca. Może się zdarzyć, że pod koniec projektu, gdy zachodzi potrzeba scalenia zmian wszystkich zespołów rozwojowych, okaże się, że nic nie działa. W efekcie uzyskujemy kod, który się nie buduje, i nie przechodzą żadne testy. Rozwiązanie każdego problemu wymaga wielu dni lub tygodni prowadzenia dochodzenia w celu ustalenia, co spowodowało awarię i w jaki sposób trzeba ją naprawić. Dodatkowym efektem jest słaby poziom obsługi klientów.

## IDEALNE ŚRODOWISKO DEVOPS. CZAS REALIZACJI WDRAŻANIA RZĘDU MINUT

W idealnym środowisku DevOps deweloperzy otrzymują szybkie i stałe sprzężenie zwrotne dotyczące ich pracy, co pozwala im szybko i samodzielnie implementować, integrować i walidować swój kod oraz wdrażać go w środowisku produkcyjnym (samodzielnie albo przez innych).

Można to osiągnąć przez iteracyjne wprowadzanie do repozytorium kontroli wersji niewielkich zmian, wykonywanie dla nich zautomatyzowanych testów badawczych oraz wdrażanie ich do produkcji. Dzięki temu można uzyskać wysoki stopień zaufania, że wprowadzone zmiany będą działać w środowisku produkcyjnym zgodnie z przeznaczeniem, a wszelkie problemy zostaną szybko wykryte i rozwiązane.

Najłatwiej to osiągnąć, gdy mamy architekturę, która jest modułowa, dobrze zhermetyzowana i preferuje luźne powiązania. Dzięki temu niewielkie zespoły mogą pracować w warunkach wysokiego stopnia autonomii, a awarie są niewielkie, dotyczą zamkniętego fragmentu i nie powodują globalnych zakłóceń.

W takim scenariuszu czas realizacji wdrożenia jest mierzony w minutach lub w najgorszym przypadku w godzinach. Uzyskana mapa strumienia wartości powinna mieć postać podobną do przedstawionej na rysunku 4:



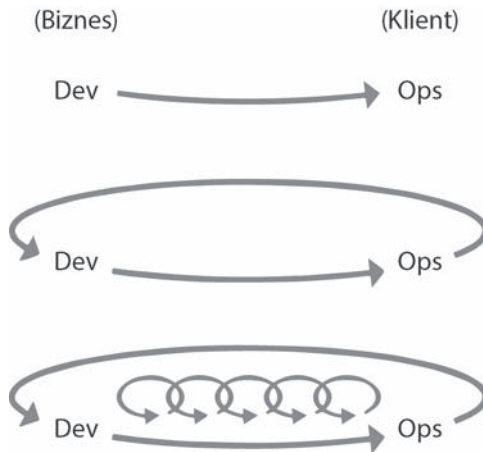
Rysunek 4. Strumień wartości technologii z czasem realizacji rzędu minut

## OBSERWACJA WSKAŹNIKA %C/A JAKO METRYKI KONIECZNOŚCI WYKONYWANIA PRZERÓBEK

Trzecią kluczową metryką w strumieniu wartości technologii oprócz czasów realizacji i przetwarzania jest wskaźnik procentu *C/A* (ang. *complete and accurate* — dosł. „gotowe i dokładne”). Metryka ta odzwierciedla jakość wyników każdego etapu strumienia wartości. Karen Martin i Mike Osterling opisali tę metrykę w następujący sposób: „Wartość wskaźnika % *C/A* można uzyskać, zadając klientom dalszego etapu strumienia pytanie o procent przypadków, gdy otrzymują pracę w postaci gotowej do wykorzystania, co oznacza, że mogą oni wykonywać swoją pracę bez konieczności korygowania dostarczonych informacji, dodawania brakujących danych lub wyjaśniania sformułowań, które mogłyby i powinny być czytelniejsze”.

# TRZY DROGI. ZASADY LEŻĄCE U PODSTAW DEVOPS

W *Projekcie Feniks* zaprezentowano trzy drogi jako zbiór podstawowych zasad, z których wywodzą się wszystkie obserwowane zachowania i wzorce DevOps (rysunek 5).



**Rysunek 5.** Trzy drogi (źródło: Gene Kim, „The Three Ways: The Principles Underpinning DevOps”, blog IT Revolution Press, 9 sierpnia 2016, <http://itrevolution.com/the-three-ways-principles-underpinning-devops/>)

Pierwsza droga umożliwia szybki przepływ pracy od lewej do prawej, od działu rozwoju, poprzez dział operacji, do klienta. Aby zmaksymalizować przepływ, należy spowodować, aby praca była widoczna, zmniejszyć rozmiary partii i interwały pracy, tworzyć jakość poprzez przeciwdziałanie przekazywania defektów do centrów pracy w dół strumienia oraz stale optymalizować pracę, by osiągać globalne cele.

Dzięki przyspieszeniu przepływu przez strumień wartości technologii możemy skrócić czas realizacji zleceń wewnętrznych lub żądań klientów — zwłaszcza czas potrzebny do wdrożenia kodu w środowisku produkcyjnym. Dzięki temu podnosimy jakość pracy, jak również przepustowość i zwiększamy szanse na pokonanie konkurencji.

Praktyki stosowane do osiągnięcia tego celu obejmują procesy ciągłego budowania, integracji, testowania i wdrażania, tworzenie środowisk na żądanie, ograniczanie pracy w toku (WIP) oraz budowanie systemów i organizacji, które zapewniają bezpieczeństwo zmian.

Druga droga umożliwia szybki i stały przepływ informacji zwrotnych od prawej do lewej na wszystkich etapach strumienia wartości. Wzmocnienie sprzężenia zwrotnego jest potrzebne do przeciwdziałania wystąpieniu podobnych problemów w przyszłości lub wspomoczenia procesu ich wykrywania oraz eliminowania skutków. W ten sposób tworzymy jakość u źródła i generujemy albo „osadzamy” wiedzę tam, gdzie jest ona potrzebna. To pozwala tworzyć coraz bezpieczniejsze systemy pracy, gdzie

problemy są znajdowane i rozwiązywane wcześniej, zanim wywołają katastrofalne skutki.

Dzięki dostrzeżeniu problemów natychmiast po ich wystąpieniu oraz dzięki ich gromadzeniu do czasu znalezienia skutecznych środków zaradczych stale skracamy i wzmacniamy pętle sprzężeń zwrotnych — najważniejszy mechanizm prawie wszystkich nowoczesnych metodologii usprawniania procesów. To maksymalizuje możliwości uczenia się i doskonalenia organizacji.

Trzecia droga polega na stworzeniu generatywnej kultury wysokiego zaufania, która pozwala na dynamiczne, zdyscyplinowane i naukowe podejście do eksperymentowania i podejmowania ryzyka. To wspomaga proces czerpania nauki — zarówno z sukcesów, jak i porażek. Ponadto dzięki ciągłemu skracaniu i wzmacnianiu pętli sprzężenia zwrotnego tworzymy coraz bezpieczniejsze systemy pracy. Jesteśmy lepiej przygotowani na podejmowanie ryzyka i eksperymentowanie, co pomaga uczyć się szybciej niż konkurencja i zwyciężać na rynku.

W ramach trzeciej drogi projektujemy również nasz system pracy w taki sposób, aby można było pomnożyć efekty nowej wiedzy — przekształcić lokalne odkrycia na globalne usprawnienia. Bez względu na to, gdzie ktoś wykonuje swoją pracę, robi to z wykorzystaniem coraz większego, kolektywnego doświadczenia wszystkich osób w organizacji.

## PODSUMOWANIE

W tym rozdziale opisaliśmy pojęcia strumieni wartości oraz czasu realizacji jako jedne z kluczowych metryk skuteczności zarówno w strumieniu wartości produkcji, jak i technologii. Zapoznaliśmy się także z wysokopoziomowymi pojęciami każdej z trzech dróg — zasad, które leżą u podstaw DevOps.

W kolejnych rozdziałach opiszemy te zasady szczegółowo. Pierwszą jest **przepływ**. Należy dążyć do stworzenia szybkiego przepływu pracy w każdym strumieniu wartości — niezależnie od tego, czy dotyczy to produkcji, czy technologii. Praktyki umożliwiające szybki przepływ zostaną opisane w części III.



# SKOROWIDZ

## A

analiza  
  dynamiczna, 349  
  post-mortem, 306, 309  
  statyczna, 349  
analizowanie telemetrii, 245  
antykruchłość, 75  
archetypy architektoniczne, 212  
architektura, 55, 107  
  ewolucyjna, 214  
  luźnych sprzężeń, 118  
  systemu, 267  
ATDD, 165  
ATO, authority to operate, 355  
audytor, 371  
automatyczne  
  skalowanie, 251  
  testy, 160  
  zmiany infrastruktury, 367  
automatyzacja, 183  
  procesu wdrożeń, 185  
  standardowych procesów, 319  
awaria, 306  
  potoku wdrożeń, 169  
awarie produkcyjne, 313

## B

badania niszczące, 368  
baza danych zarządzania konfiguracją, 146  
bezpieczeństwo, 70, 211  
  aplikacji, 349  
  informacji, 343, 357  
  łańcucha dostaw oprogramowania, 353  
  pracy, 118  
  środowiska, 355  
  wdrażania kodu, 259  
bezpieczne wdrożenia, 261  
bezpieczny system pracy, 61  
biblioteka DML, 146  
błędy, 162  
budowanie nowej wiedzy, 62

## C

C/A, complete and accurate, 43  
CAB, change advisory board, 364  
CDE, Cardholder Data Environment, 370  
cele organizacji, 325  
chat roomy, 317  
ciągła integracja, 173, 178, 179

ciągłe  
budowanie, 156  
dostarczanie, 139, 206  
uczenia się, 32, 299  
wdrażanie, 206  
coaching kata, 76  
COTS, commercial off-the-shelf, 391  
czas  
przetwarzania, 41  
realizacji wdrażania, 43  
czasy realizacji wdrażania, 42

## D

DAA, Designated Approving Authority, 355  
definiowanie czasu realizacji, 41  
dług techniczny, 101, 330  
DML, Definitive Media Library, 146  
dni gier, 314  
dodatkowe  
funkcje, 56  
procesy, 56  
dokumentacja, 322, 371  
dostarczanie wartości do klienta, 93  
dostęp do telemetrii, 236  
druga droga, 59  
dzielenie się doświadczeniami, 334  
dzień gry, 75

## E

eksperymentowanie, 32, 69, 299  
eliminowanie  
ograniczeń, 54  
procesów biurokratycznych, 295  
trudności, 55  
etyka Devops, 25

## F

finansowanie, 117

## G

gated commits, 178  
generalisci, 116

globalne usprawnienia, 317  
GRC, governance, risk, and compliance, 345  
grouplet, 336

## H

heroizm, 57  
higiena produkcji, 267  
historijki użytkownika, 324  
HRR, Hand-Off Readiness Review, 270

## I

IDE, integrated development environment, 158  
idealne środowisko Devops, 43  
identyfikowanie ograniczeń, 54  
implementacja przełączników funkcji, 202  
infrastruktura, 147  
telemetrii, 228  
innowacje, 89  
innowatorzy, 90  
instalacje kodu, 54  
instrumentacja, 248  
środowiska, 359  
integracja  
mechanizmów zabezpieczeń, 363  
technik wytwarzania oprogramowania, 273  
zabezpieczeń, 347  
zadań wdrażania, 190  
integralność kodu źródłowego, 350  
integrowanie  
inżynierów OPS, 131  
kodu, 156  
środowisk, 156  
inżynieria odporności, 314  
inżynierowie  
DEV, 263  
OPS, 263

## J

jakość, 64



## K

kaizen blitzes, 73  
kolejki, 388  
komentarze, 282  
kompilacja, 190  
konfiguracja testów, 54  
konflikty, 20  
kontrola  
    jakości, 65  
    zmian, 284  
koordynacja, 281, 286  
koszty, 24  
kultura uczenia się, 70, 76

## L

Lean, 383  
liczba  
    defektów, 267  
    przełączeń, 53  
limit WIP, 51  
limity pracy w toku, 50  
linka Andon, 170, 391  
LRR, launch readiness review, 270  
luki w telemetrii, 239  
luźno sprzężone usługi, 119

## M

małpia armia, 394  
manifest Agile, 384  
mapa strumienia wartości, 93, 96  
marketing  
    bezpośredni, 275  
    masowy, 275  
mechanizm  
    kontroli zabezpieczeń, 346  
    kontroli zmian, 284  
    odsyłania usługi, 268  
menedżer strumienia wartości, 95  
menedżerowie  
    techniczni, 95  
    wydania, 95  
metryka konieczności wykonywania  
    przeróbek, 43

metryki  
    biznesowe, 240  
    infrastruktury, 242  
    poziomu aplikacji, 240  
    produkcji, 234  
mikrousługi, 212  
mity, 11, 390  
model  
    eksperymentalny, 311  
    standardowy, 311  
monolity, 212  
MVP, minimal viable product, 330

## N

narzędzia, 288  
natywna chmura, 304  
norma PCI, 369

## O

obliczanie średnich, 246  
obserwator, 291  
ocena efektywności procesów, 293  
ochrona potoku wdrożeń, 360, 363  
oczekiwanie, 56  
odchylenie standardowe, 246  
oddzielenie wdrożeń od wydań, 194  
ograniczenie pracy, 49  
oprogramowanie bazujące na rewizji  
    master, 175  
oprogramowanie  
    COTS, 391  
    sterowane hipotezami, 273  
optymalizacja, 66  
    kosztów, 111  
    szybkości, 112  
organizacje  
    funkcjonalne, 110  
    macierzowe, 110  
    rynkowe, 110  
orientacja funkcjonalna, 113  
OWASP, 349

## P

- pierwsza droga, 47
- pionierzy, 90
- planowanie
  - analiz post-mortem, 306
  - cech funkcjonalnych, 278
  - poprawy, 100
  - zmian, 286
- podejście
  - big bang, 149
  - rozwojowe, 117
  - sztywne, 117
- podział obowiązków, 369
- poprawa widoczności pracy, 105
- poprawka w przód, 262
- POS, point of sale, 330
- potok wdrożeń, 141, 169, 190, 347, 363
- poziom
  - aplikacji, 239, 248
  - bazy danych, 248
  - biznesowy, 239
  - DEBUG, 232
  - ERROR, 232
  - FATAL, 232
  - INFO, 232
  - infrastruktury, 239
  - oprogramowania klienckiego, 239
  - potoku wdrożeń, 239
  - sieci, 248
  - systemu operacyjnego, 248
  - WARN, 232
- praca parami, 291
- prace
  - niestandardowe, 57
  - wykonane częściowo, 56
- praktyki UX, 265
- prawo Conwaya, 107, 118
- prezentowanie realizacji zadań, 344
- problemy, 20, 61
- proces
  - wdrażania, 267
  - zatwierdzania zmian, 283
- produktywne utrzymanie ruchu, 383
- program
  - Hubot, 318
  - NR, 74

- programowanie
  - sterowane testami, 291
  - w parach, 288, 291, 292
- projektowanie
  - granic zespołów, 118
  - organizacji, 107
- promienniki informacji, 236
- przeгляд, 281
  - zmian, 287
- przełączanie
  - pomiędzy zadaniami, 56
  - pracy, 388
- przełączenia, 53
- przełącznik cech funkcjonalnych, 202
- przenoszenie pracy, 48
- przepływ, 32, 45, 47
  - pierwsza droga, 47
- przewlekłe konflikty, 386
- przezroczysty czas sprawności, 395
- przydzielenie łączników ops, 130
- publikowanie, 277

## R

- raport DBIR, 354
- rejestrowanie zdarzeń, 231
- repozytorium
  - kodu źródłowego, 289, 320, 346
  - prawdy, 145
- retrospektywy, 133
- rewizja master, 175–178
- rewizje, 145
- RFC, request for change, 364
- router zdarzeń, 229
- rozkład Gaussa, 247, 249
- rozmiar
  - partii, 51
  - zespołów, 120
- rozpowszechnianie
  - praktyk, 336
  - wiedzy, 322
- rozwiązywanie problemów, 62, 225, 233
- rozwijanie metodyki, 91
- ruch, 56
  - Agile, 384
  - ciągłych dostaw, 385
  - konferencji Velocity, 384

Lean, 383  
Lean Startup, 385  
Lean UX, 386  
Rugged Computing, 386  
Toyota Kata, 385  
RUP, rational unified process, 384

## S

scrum, 132  
SDLC, system development life cycle, 369  
silosy, 115  
skalowanie  
    wydajności, 29  
    zależności, 350  
skuteczność orientacji funkcjonalnej, 113  
słabe sygnały awarii, 311  
SOA, service-oriented architectures, 119  
SOE, system of engagement, 86  
SOR, system of record, 86  
SOT, shared operations team, 187  
specjaliści, 116  
spirala degradacji, 21, 25, 387  
spotkania post-mortem, 392  
sprint, 149  
sprzężenia zwrotne, 32, 59, 259  
SRE, site reliability engineers, 269  
standaryzacja stosu technologii, 326  
standupy, 132  
strangler application, 215  
strangler application pattern, 210  
strumień wartości  
    identyfikacja zespołów, 95  
    inżynier pełnego stosu, 116  
    mapa, 96  
    produkcji, 39  
    technologii, 40  
    wybór, 83  
swarming, 62  
system  
    SoE, 86, 88  
    SoR, 86, 88  
    uczenia się, 329  
systemy CIS, 200  
szybka transformata Fouriera, 252

## Ś

śledzenie defektów, 345  
ślepe uruchomienia, 203  
średnia, 247  
środowisko zbliżone do produkcyjnego, 149

## T

tablica kanban, 134  
TBD, trunk-based development, 175  
TC, Test Certified, 337  
TDD, test-driven development, 165, 184, 291  
techniczne praktyki przepływu, 137  
techniki TDD, 184  
telemetria, 225, 233, 239  
    produkcji, 357  
    zabezpieczeń, 358  
teoria  
    ograniczeń, 386  
    zgniłego jabłka, 305  
test, 54  
    A/B, 275  
    weryfikacyjny, 160  
testowalność, 211  
testowanie, 114, 156, 190  
    automatyczne, 153  
    cechy funkcjonalnej, 276  
    statycznych zabezpieczeń, 351  
    wydajności, 167  
    wymagań niefunkcjonalnych, 168  
testy  
    akceptacyjne, 161  
    integracji, 161  
    jednostkowe, 161  
    ręczne, 165  
TPM, Total Productive Maintenance, 383  
trendy, 19  
trenowanie próbných awarii, 314  
trzecia droga, 69  
tworzenie  
    danych, 198  
    infrastruktury, 147  
    metryk produkcji, 234  
    repozytorium prawdy, 145

tworzenie  
samoobsługowych metryk, 237  
środowisk na żądanie, 143  
środowiska, 54  
telemetrii, 225  
telemetrii rejestrowania zdarzeń, 231  
typ alertów, 267

## U

uczenie się, 76, 303, 305, 329, 333  
uniwersalność rozwiązania, 30  
uruchamianie testów, 54  
usługa Chaos Monkey, 394  
usługi  
brownfield, 86  
greenfield, 86  
niezmienne, 215  
wersjonowane, 215  
współdzielone, 346  
usprawnienia, 72, 317  
globalne, 74  
utrzymywanie spójnych środowisk, 187  
uzgadnianie wspólnego celu, 100

## W

wady ukryte, 315  
wartość biznesowa metodyki DevOps, 28  
wchodzenie z hukiem, 90  
wdrażanie, 40, 186, 190  
wdrożenia, 195  
automatyczne, 189  
produkcyjne, 263  
samoobsługowe, 189  
wewnętrzne konsultacje, 336  
WIP, Work In Progress, 50  
właściciel produktu, 95  
wprowadzenie inżynierów do zespołów, 129  
wskaźnik procentu C/A, 43  
współbieżność, 164  
wstrzykiwanie awarii produkcyjnych, 313  
wybór strumieni wartości, 83  
wydajność, 211  
wydanie, 195  
wyglądanie, 253

wykrywanie  
anomali, 253, 255  
błędów, 162  
odstających, 246  
wymagania niefunkcjonalne, 101  
ujednolicone, 323  
wytwarzanie oprogramowania  
małymi partiami, 177  
sterowane testami, 165  
wyuczona bezradność, 24  
wzmacnianie  
kultury uczenia się, 76  
pożądanych zachowań, 105  
wzorce  
odporności, 75  
wydań bazujące na aplikacjach, 196  
wydań bazujące na środowisku, 195  
wydań kanarkowych, 200  
wzorzec  
aplikacji dusiciela, 216  
strangler application, 210, 215  
wdrażania niebieskie-zielone, 196

## Z

zabezpieczenia, 114  
zagrożenia, 290  
zakres monitorowania, 267  
zamrażanie zmian, 290  
zarządzanie  
usługą produkcyjną, 266  
zmianami, 339  
zasada  
ciągłego uczenia się, 69  
przepływu, 47  
sprzężenia zwrotnego, 59  
zbieranie danych, 229  
zespoły rynkowe, 112  
zespół  
deweloperów, 95  
operacji współdzielonych, 187  
operacyjny, 95  
transformacji, 98  
walidacji, 95  
zabezpieczeń, 95  
zarządzania zmianami, 288

- zgodność z przepisami, 339, 363, 371
- zielona kompilacja, 160
- zintegrowane środowisko programisty, 158
- zintegrowanie zadań, 125
- złożone systemy, 60
- zmiany
  - niskiego ryzyka, 365
  - normalne, 364, 366
  - pilne, 365
  - standardowe, 364
- zmniejszenie
  - liczby przełączeń, 53
  - wielkości partii, 50
- zwiększanie wydajności pracy, 127



# PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW  
w działający bankomat!

**Dowiedz się więcej i dołącz już dzisiaj!**

<http://program-partnerski.helion.pl>



## Zwinność, niezawodność i bezpieczeństwo — w skrócie: DevOps!

Organizacje, które cechuje wysoka dynamika rozwoju, w naturalny sposób będą wprowadzać kolejne innowacje i wygrywać na rynku. Metodyka DevOps jest bezcennym narzędziem dla takich jednostek: pozwala na tworzenie bezpiecznych systemów pracy i umożliwia niewielkim zespołom szybkie i niezależne rozwijanie oraz walidację kodu, który po krótkim czasie można bezpiecznie wdrożyć u klientów. Dzięki niej szybko można uzyskać korzyści ze współdziałania wszystkich form inżynierii w sposób wielodyscyplinarny!

Niniejsza książka powinna znaleźć się na podorędziu każdego menedżera z branży IT. Obecnie sposób zarządzania technologią może stanowić o przetrwaniu firmy, więc przyjęcie nowych zasad i praktyk w tym zakresie jest kluczową decyzją. I o tym właśnie mówi ta książka. Doceni ją każdy, kto zarządza pracą w strumieniu wartości technicznych (lub w nim pracuje), który zwykle obejmuje zarządzanie produktem, rozwój, walidację, operacje IT i bezpieczeństwo informacji. Książka przyda się także menedżerom biznesowym, którzy inicjują działania w obszarze technologii.

W tej książce między innymi:

- podstawy DevOps i wysokopoziomowe zasady trzech dróg
- strumienie wartości, zasady i wzorce projektowania, wzorce wdrażania oraz studia przypadków
- jak zbudować podstawy protokołu wdrożeń
- przyspieszanie i wzmacnianie sprzężeń
- stymulowanie ciągłego uczenia się oraz kultura odpowiedzialnego traktowania się
- poprawna integracja zasad zabezpieczeń i zgodności z przepisami

**Gene Kim** jest amerykańską przedsiębiorcą, badaczem i autorem książek o prowadzeniu biznesu. Szczególnie interesuje się technologiami IT, bezpieczeństwem informacji i metodyką DevOps. **Patrick Debois** jest konsultantem i organizatorem konferencji — zorganizował pierwsze wydarzenia poświęcone DevOps. To on jest autorem nazwy tej metodyki. **John Willis** pracuje w branży IT od ponad 35 lat. Jest dyrektorem działu rozwoju w firmie Docker. Napisał sześć Czerwonych Książ IBM dotyczących systemów zarządzania w przedsiębiorstwach. **Jez Humble** od ponad 10 lat pracuje jako programista, administrator systemów, konsultant i szef. Z jego wiedzy korzystały organizacje non profit, telekomy, firmy finansowe i sklepy internetowe. **John Allspaw** od 15 lat współpracuje z firmami biotechnologicznymi, jednostkami administracji rządowej i mediami online. Jest twórcą infrastruktury wspierającej takie serwisy jak Salon, InfoWorld czy Flickr.

sięgnij po WIĘCEJ!



KOD KORZYSCI

**Helion**

Księgarnia Internetowa



<http://helion.pl>

zamówienia telefoniczne



0 801 339900



0 601 339900

Helion SA  
ul. Kościuszki 1c, 44-100 Gliwice  
tel.: 32 230 98 63  
e-mail: [helion@helion.pl](mailto:helion@helion.pl)  
<http://helion.pl>

Sprawdź najnowsze promocje:  
● <http://helion.pl/promocje>  
Książki najchętniej czytane:  
● <http://helion.pl/bestsellery>  
Zamów informacje o nowościach:  
● <http://helion.pl/nowosci>

ISBN 978-83-283-3453-3



9 788328 334533

cena: 77,00 zł

onepress

REVOLUTION

Informatyka w najlepszym wydaniu