Wprowadzenie do DevOps i współczesnego administrowania systemów

Zarządzanie Systemami Rozproszonymi

mgr inż. Jakub Woźniak

Politechnika Poznańska
Wydział Informatyki i Telekomunikacji

jakub.wozniak@cs.put.poznan.pl
www.cs.put.poznan.pl/jwozniak

Agenda wykładu

1. Wprowadzenie i cele przedmiotu
2. Ewolucja roli administratora systemów
3. Podstawy filozofii DevOps
4. Współczesne wyzwania i trendy IT 2025
5. Case Study: Santander Bank Polska
6. Podsumowanie

O mnie

- **Jakub Woźniak** - CTO, Software Services w Nordcloud, an IBM Company
- Doświadczenie w architekturze systemów rozproszonych i cloud computing
- Praktyczne wdrożenia DevOps w organizacjach enterprise
- Specjalizacja w architekturze serverless, cloud-native, kubernetes
- AWS, Azure, Google Cloud

Kontakt

jakub.wozniak@cs.put.poznan.pl
www.cs.put.poznan.pl/jwozniak

Cele przedmiotu

Cel główny

Zapoznanie z nowoczesnymi praktykami zarządzania systemami rozproszonymi ze szczególnym uwzględnieniem metodologii DevOps, konteneryzacji oraz zadań współczesnego administratora systemów.

Cele szczegółowe

1. Zrozumienie fundamentów filozofii DevOps
2. Opanowanie koncepcji konteneryzacji i orkiestracji
3. Poznanie narzędzi do automatyzacji infrastruktury
4. Zrozumienie monitorowania i obserwabilności
5. Poznanie strategii backup i odzyskiwania danych
6. Zrozumienie wyzwań bezpieczeństwa w architekturach rozproszonych

Struktura kursu - 10 wykładów

Część I: Fundamenty

1. Wprowadzenie do DevOps
2. Konteneryzacja - Docker
3. Kubernetes podstawy
4. Infrastructure as Code
5. CI/CD Pipeline

Część II: Operacje

6. Monitorowanie systemów
7. DevSecOps
8. Backup i Recovery
9. Chmura i skalowanie
10. AI/ML w DevOps

Dlaczego DevOps?

87% organizacji używa strategii multi-cloud, rynek chmury wzrośnie z $483B (2022) do $1.6T (2030)

Timeline ewolucji administratora

- **1990-2000:** Fizyczne serwery, reaktywne utrzymanie
- **2000-2010:** VMware, centralne zarządzanie
- **2010-2025:** Cloud-first, Infrastructure as Code, automatyzacja

Tradycyjny model (1990-2000)

Charakterystyka roli:

- Zarządzanie fizycznymi serwerami
- Instalacja i konfiguracja OS
- Backup na taśmach i płytach
- Reaktywne rozwiązywanie problemów
- Podział: "development vs operations"

Typowe narzędzia:

- Skrypty powłoki
- Zadania Cron
- Monitorowanie SNMP
- Ręczne wdrożenia
- Dokumentacja w Excel

Problem

Brak automatyzacji, długie czasy wdrożenia, wysokie ryzyko błędów ludzkich, słaba skalowalność

Era wirtualizacji (2000-2010)

Wpływ VMware:

- Maksymalizacja wykorzystania sprzętu
- Łatwiejsze zarządzanie zasobami
- Pierwsze kroki automatyzacji
- Scentralizowane zarządzanie (vCenter)

Nowe wyzwania:

- Złożoność środowisk wirtualnych
- Potrzeba nowych umiejętności
- Sieciowe przechowywanie danych (SAN/NAS)
- Kopie zapasowe maszyn wirtualnych

Kluczowa zmiana

Przejście od zarządzania sprzętem do zarządzania abstrakcją - pierwsza lekcja Infrastructure as Code

Rewolucja chmurowa (2010-2025)

Transformacja AWS/Azure/GCP:

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Serverless computing
- Pay-per-use model
- Global scale w minuty

Nowe role:

- Cloud Architect
- DevOps Engineer
- Site Reliability Engineer (SRE)
- Platform Engineer
- DevSecOps Specialist

Kluczowe zmiany umiejętności

Od "click-click" do "code-code" - API-first approach, Infrastructure as Code, monitoring i observability

Statystyki transformacji

Wzrost danych globalnych:

- 2020: 64.2 ZB
- 2025: 180 ZB (przewidywane)
- Wzrost o 180% w 5 lat

Rynek chmury:

- 2022: $483B
- 2030: $1.6T (przewidywane)
- CAGR: 15.7%

Adopcja multi-cloud:

- 87% organizacji (2024)
- Średnio 2.6 cloud providers
- Hybrid cloud: 65% enterprise

Adopcja DevOps:

- 83% organizacji IT
- 200x częstsze wdrożenia
- 24x szybsze odzyskiwanie

Co to jest DevOps?

Definicja

DevOps = Development + Operations = metodologia skracająca cykl SDLC przy jednoczesnym zapewnieniu wysokiej jakości oprogramowania

Historia powstania:

- 2009: Patrick Debois - "DevOpsDays" Belgia
- John Allspaw & Paul Hammond - "10+ Deploys Per Day" (Flickr)
- Odpowiedź na problemy komunikacji dev-ops

Kluczowe założenia:

- Szybsze dostarczanie wartości biznesowej
- Większa niezawodność systemów
- Redukcja ryzyka przy wdrożeniach
- Lepsza współpraca zespołów

"You build it, you run it" - Amazon

Zespoły deweloperskie biorą pełną odpowiedzialność za cały cykl życia aplikacji

Framework CALMS

C.A.L.M.S

C	A	L	M	S
Culture	**Automation**	**Lean**	**Measurement**	**Sharing**
Kultura	Automatyzacja	Szczupłość	Pomiary	Dzielenie się

Holistyczne podejście

DevOps to nie tylko narzędzia - to transformacja kulturowa, procesowa i technologiczna jednocześnie

C - Culture (Kultura)

Wspólna odpowiedzialność:

- Wspólna odpowiedzialność za produkt
- Koniec z "nie mój problem"
- Własność od początku do końca

Bezpieczeństwo psychologiczne:

- Prawo do błędu i uczenia się
- Analizy bez przypisywania winy
- Kultura eksperymentowania

Łamanie silosów:

- Likwidacja barier organizacyjnych
- Zespoły międzyfunkcyjne
- Wspólne cele i metryki

Ciągłe uczenie się:

- Inwestycja w rozwój zespołu
- Sesje dzielenia się wiedzą
- Społeczności praktyków

Przykład: Netflix

"Freedom and Responsibility" culture - autonomia zespołów przy jasnych oczekiwaniach biznesowych

A - Automation (Automatyzacja)

Pipelines CI/CD:

- Automatyczne budowanie
- Automatyczne testowanie
- Automatyczne wdrażanie
- Mechanizmy cofania zmian

Infrastruktura jako kod:

- Terraform, Ansible, Pulumi
- Wersjonowana infrastruktura
- Powtarzalne środowiska

Automatyzacja testów:

- Testy jednostkowe
- Testy integracyjne
- Testy end-to-end
- Testy wydajnościowe

Systemy samoleczące:

- Automatyczne skalowanie
- Kontrole zdrowia
- Automatyczne odzyskiwanie
- Wyłączniki bezpieczeństwa

Przykład: Facebook

1000+ wdrożeń dziennie dzięki pełnej automatyzacji potoków

L - Lean (Szczupłość)

Minimalizacja marnotrawstwa:

- Eliminacja marnotrawstwa
- Skupienie na działaniach dodających wartość
- Redukcja przełączania kontekstu

Małe partie:

- Częste, małe zmiany
- Ograniczony zasięg błędów
- Szybsze pętle informacji zwrotnej

Mapowanie strumienia wartości:

- Optymalizacja przepływu wartości
- Identyfikacja wąskich gardeł
- Skrócenie czasu realizacji

Szybko błądzić, szybko uczyć się:

- Szybkie wykrywanie błędów
- Szybkie prototypowanie
- Iteracyjne doskonalenie

Kluczowa zasada

Optymalizacja całego systemu, nie pojedynczych komponentów

M - Measurement (Pomiary)

Metryki techniczne:

- MTTR (średni czas odzyskiwania)
- MTBF (średni czas między awariami)
- Częstotliwość wdrożeń
- Czas realizacji zmian

Metryki biznesowe:

- Zadowolenie klientów
- Wpływ na przychody
- Wskaźnik adopcji funkcji
- Zaangażowanie użytkowników

Obserwowalność:

- Metryki, logi, ślady
- Monitorowanie w czasie rzeczywistym
- Strategie alertów
- Pulpity i wizualizacja

Decyzje oparte na danych:

- Testy A/B
- Przełączniki funkcji
- Testy porównawcze wydajności
- Planowanie pojemności

Pamiętaj

"Nie można ulepszyć tego, czego się nie mierzy" - ciągłe monitorowanie wszystkich aspektów systemu

S - Sharing (Dzielenie się)

Dzielenie się wiedzą:

- Dokumentacja jako kod
- Wewnętrzne wiki i podręczniki
- Regularne sesje szkoleniowe
- Prezentacje techniczne

Standaryzacja narzędzi:

- Wspólne narzędzia i praktyki
- Współdzielone biblioteki i komponenty
- Standardowe procedury operacyjne

Zespoły międzyfunkcyjne:

- Zespoły międzyfunkcyjne
- Wbudowani inżynierowie operacyjni
- Wspólna odpowiedzialność za dyżury

Otwarta komunikacja:

- Transparentność procesów
- Regularne retrospektywy
- Dzielenie się incydentami i uczenie

Community Building

Tworzenie wewnętrznych społeczności praktyków - silosy znikają, wiedza się rozprzestrzenia

Korzyści DevOps - dane z State of DevOps Report 2024

Metryki wydajności:

- **200x** częstsze wdrożenia
- **24x** szybsze odzyskiwanie po incydentach
- **3x** niższa częstotliwość błędów
- **50%** mniej czasu na poprawki

Wpływ biznesowy:

- **23%** wzrost rentowności
- **19%** wzrost udziału w rynku
- **18%** wzrost produktywności
- **12%** wzrost zadowolenia klientów

Elite Performers

Najlepsze 25% organizacji osiąga wszystkie powyższe metryki jednocześnie - DevOps jako przewaga konkurencyjna

Wyzwania implementacji DevOps

Organizacyjne:

- Opór kulturowy - "zawsze tak robiliśmy"
- Mentalność silosów - zachowania terytorialne
- Brak wsparcia kierownictwa
- Niewystarczający budżet szkoleniowy

Techniczne:

- Systemy dziedziczone - dług techniczny
- Rozproszenie narzędzi - za dużo narzędzi
- Obawy bezpieczeństwa - luka DevSecOps

Anti-patterns do unikania:

- "DevOps team" jako kolejne silos
- Automatyzacja bez zmiany kultury
- Pomijanie testów w pogoni za szybkością
- Brak monitorowania w produkcji
- "DevOps engineer" zamiast wspólnej odpowiedzialności

Kluczowy insight

Transformacja DevOps = 20% narzędzia, 80% kultura i procesy

AI/ML Revolution w IT Operations

AIOps - Artificial Intelligence for IT Operations

Zastosowanie uczenia maszynowego do automatyzacji i optymalizacji operacji IT

Praktyczne zastosowania:

- **Analityka predykcyjna** - konserwacja zapobiegawcza
- **Wykrywanie anomalii** - automatyczne wykrywanie incydentów
- **Inteligentna alokacja zasobów** - optymalizacja kosztów
- **Rozpoznawanie wzorców** - analiza logów i przyczyn źródłowych

Narzędzia i platformy:

- Datadog Intelligence
- Microsoft Azure Monitor
- Google Cloud Operations Suite
- Splunk IT Service Intelligence
- Dynatrace Davis AI

Wpływ na role

Przejście od operacji reaktywnych do predykcyjnych - nowe umiejętności: podstawy nauki o danych, interpretacja modeli ML

Platform Engineering - nowy paradygmat

Platform Engineering

Dyscyplina projektowania i budowy łańcuchów narzędziowych i przepływów pracy, które umożliwiają samoobsługowe możliwości dla inżynierów oprogramowania w erze cloud-native

Wewnętrzna platforma deweloperska (IDP):

- Samoobsługowa infrastruktura
- Złote ścieżki i szablony
- Optymalizacja doświadczenia dewelopera
- Standardowe wzorce wdrażania

Korzyści:

- Obniżone obciążenie poznawcze deweloperów
- Szybszy czas wprowadzenia do produkcji
- Zwiększona produktywność deweloperów
- Spójne bezpieczeństwo i zgodność

Przykłady platform

Spotify Backstage, Netflix Spinnaker, Uber Peloton, Google Borg

Edge Computing i distributed infrastructure

Czynniki napędzające Edge Computing:

- Proliferacja sieci 5G
- Eksplozja urządzeń IoT
- Wymagania niskich opóźnień
- Regulacje lokalności danych (GDPR)
- Optymalizacja kosztów przepustowości

Wzorce architektoniczne:

- Kubernetes na brzegu (k3s, MicroK8s)
- Ewolucja CDN
- Fog computing
- Wielodostępowe przetwarzanie brzegowe (MEC)

Wyzwania dla administratorów:

- Zarządzanie tysiącami węzłów brzegowych
- Bezpieczeństwo w środowiskach rozproszonych
- Monitorowanie na masową skalę
- Niezawodność i łączność sieci
- Automatyczne udostępnianie i aktualizacje

Nowe umiejętności:

- Platformy orkiestracji brzegowej
- Programowanie sieciowe
- Debugowanie systemów rozproszonych

Sustainability i Green IT

Green Computing Imperative

Europejski Zielony Ład i regulacje ESG wymuszają optymalizację śladu węglowego w operacjach IT

Optymalizacja śladu węglowego:

- Energooszczędne planowanie obciążeń
- Wybór zielonych regionów chmury
- Właściwe wymiarowanie i optymalizacja zasobów
- Śledzenie wykorzystania energii odnawialnej
- Narzędzia oceny cyklu życia

Praktyczne działania:

- Autoskalowanie uwzględniające emisję CO2
- Zrównoważony rozwój oprogramowania
- Zielone pipelines CI/CD
- Monitorowanie i raportowanie energii
- Gospodarka cyrkularowa w sprzęcie IT

Regulatory landscape

Taksonomia UE, Dyrektywa o raportowaniu zrównoważoności przedsiębiorstw (CSRD) - zgodność staje się obowiązkowa

Zero Trust Security Architecture

"Never trust, always verify"

Model bezpieczeństwa oparty na ciągłej weryfikacji zamiast ochrony perymetrycznej

Główne zasady:

- Model bezpieczeństwa zorientowany na tożsamość
- Dostęp z najmniejszymi uprawnieniami
- Mikrosegmentacja
- Ciągłe uwierzytelnianie
- Założenie naruszenia

Obszary implementacji:

- Bezpieczeństwo sieci (SASE)
- Zarządzanie tożsamością i dostępem
- Bezpieczeństwo urządzeń i aplikacji

Integracja DevSecOps:

- Bezpieczeństwo wbudowane w CI/CD
- Polityki jako kod
- Automatyczne sprawdzanie zgodności
- Automatyzacja testów bezpieczeństwa
- Automatyzacja reakcji na incydenty

Ekosystem narzędzi:

- HashiCorp Boundary
- Okta Workforce Identity
- Palo Alto Prisma
- Microsoft Conditional Access

Case Study: Santander Bank Polska - DevOps Transformation

Context

Transformacja DevOps w organizacji finansowej - zgodność regulacyjna + innowacje cyfrowe

Sytuacja wyjściowa (2012):

- Dziedziczone narzędzia zarządzania projektami
- Manualne procesy wdrażania
- Długie cykle wydań
- Słaba komunikacja zespołów dev-ops
- Wyzwania zgodności regulacyjnej

Czynniki biznesowe:

- Konkurencja z fintechami
- Oczekiwania klientów - bankowość cyfrowa
- Wymagania regulacyjne (PCI-DSS, GDPR)
- Presja optymalizacji kosztów
- Przyspieszenie czasu wprowadzenia na rynek

Kluczowe wyzwanie

Jak połączyć zwinność DevOps z surowymi wymaganiami zgodności w branży finansowej?

Faza 1: Process Management (2012-2013)

Implementowane rozwiązania:

- Migracja z dziedziczonych narzędzi do **Atlassian Jira**
- Integracja z CMDB i zarządzaniem zmianami
- Ulepszone śledzenie błędów i zarządzanie wydaniami
- Raportowanie online na wszystkich poziomach
- Zautomatyzowany przepływ pracy dla zatwierdzeń

Napotkane wyzwania:

- Opór użytkowników przed adopcją
- Złożoność migracji danych
- Integracja z istniejącymi systemami
- Szkolenia i zarządzanie zmianą
- Dostosowanie do procesów bankowych

Key insight

Adopcja narzędzi bez standaryzacji procesów = ograniczony wpływ. Zmiana kultury równie ważna jak technologia.

Faza 2: Automation (2015-2016)

Implementacja automatyzacji:

- **JFrog Artifactory** - scentralizowane zarządzanie artefaktami
- Zautomatyzowane pipelines wdrożeniowe
- Integracja z **IBM UrbanCode Deploy**
- Standaryzacja repozytoriów pakietów
- Integracja automatycznych testów

Stos techniczny:

- **Jenkins** - automatyzacja CI/CD
- **BMC Remedy** - integracja zarządzania zmianami
- Niestandardowe skrypty do wdrożeń
- Automatyzacja migracji baz danych
- Automatyzacja udostępniania środowisk

Banking-specific considerations

Automatyczne ślady audytu, procedury cofania, przepływy zatwierdzeń regulacyjnych - automatyzacja zgodności

Rezultaty i wnioski

Osiągnięte korzyści:

- **70%** skrócenie czasu wdrożenia
- **Poprawiona** stabilność systemu
- **Lepsza** śledzalność i zgodność
- **Większa** współpraca zespołowa
- **Mniej** błędów manualnych

Kluczowe czynniki sukcesu:

- **Wsparcie zarządu** - zaangażowanie kadry kierowniczej
- **Podejście stopniowe** - wdrożenie etapami
- **Inwestycja w szkolenia** - rozwój kompetencji zespołu
- **Najpierw optymalizacja, potem automatyzacja**
- **Integracja zgodności** - bezpieczeństwo od początku

Critical lesson

W branży finansowej: zgodność nie jest ograniczeniem dla DevOps - to wymaganie, które może być zautomatyzowane

Uniwersalne wnioski dla polskich firm

Co się sprawdza w polskim kontekście?

- **Stopniowa transformacja** - nie podejście "big bang"
- **Rozwiązania hybrydowe** - integracja z systemami dziedziczonymi
- **Mentalność zgodności na pierwszym miejscu** - szczególnie w regulowanych branżach
- **Inwestycja w szkolenia zespołów** - przekwalifikowanie istniejącej kadry
- **Standaryzacja narzędzi** - redukcja złożoności, poprawa utrzymywalności

Inne polskie historie sukcesu

- **Allegro** - transformacja mikroserwisów na Kubernetes
- **CD Projekt** - pipelines CI/CD w tworzeniu gier
- **Asseco** - inżynieria platform dla klientów finansowych

Kluczowe wnioski z wykładu

1. Rola administratora ewoluuje

Od reaktywnego utrzymania do proaktywnej inżynierii - Infrastruktura jako kod, myślenie automation-first

2. DevOps to kultura, nie narzędzia

CALMS framework jako przewodnik transformacji - Culture, Automation, Lean, Measurement, Sharing

3. AI i automatyzacja zmieniają landscape

AIOps, Platform Engineering, Edge Computing - nowe umiejętności są kluczowe dla rozwoju kariery

4. Polskie firmy skutecznie implementują DevOps

Santander, Allegro - przykłady do naśladowania, skupienie na zgodności i podejściu stopniowym

Następne kroki w kursie

Wykład 2 (następny tydzień):

- **Konteneryzacja** - Docker fundamenty
- Nowoczesne praktyki 2025
- Bezpieczeństwo kontenerów
- Budowy wieloetapowe
- Alternatywy: Podman, containerd

Zalecana lektura:

- "The Phoenix Project" - rozdziały 1-5
- State of DevOps Report 2024 - streszczenie kierownicze
- Przewodnik najlepszych praktyk Docker
- CNCF Landscape overview

Kontakt i zasoby

Prowadzący:
mgr inż. Jakub Woźniak

Kontakt:
jakub.wozniak@cs.put.poznan.pl
www.cs.put.poznan.pl/jwozniak

Dodatkowe zasoby:

- CNCF Landscape: landscape.cncf.io
- State of DevOps Reports
- Kubernetes Documentation
- Docker Best Practices Guide
- AWS/Azure/GCP Documentation

Community:

- DevOps Poland Meetup
- Kubernetes Poland
- Cloud Native Computing Foundation

Dziękuję za uwagę!

Pytania?

Jakub Woźniak
jakub.wozniak@cs.put.poznan.pl