



# Programowanie aplikacji w chmurze obliczeniowej

## Program przedmiotu

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni



**Fundusze  
Europejskie**  
Wiedza Edukacja Rozwój



**Rzeczpospolita  
Polska**

**Unia Europejska**  
Europejski Fundusz Społeczny



# Program przedmiotu (1)

- Pojęcie chmury, rodzaje chmur, koncepcja usług w chmurze.
- Wykorzystanie technik wirtualizacji w chmurach komputerowych.
- Obsługa pamięci masowych w chmurze komputerowej.
- Rozwiązania chmurowe w systemach Linux.
- Paradygmat programowania równoległego MapReduce, system Hadoop jako przykład programowania w chmurze.

# Program przedmiotu (2)

- Elementy aplikacji w chmurze komputerowej.
- Wykorzystanie języka Java do programowania w chmurze komputerowej.
- Usługi Pig i Hive jako elementy chmury komputerowej.
- Chmura komputerowa Amazon.
- Rozwiązania chmurowe firm Google oraz Microsoft.

# Efekty uczenia się w zakresie wiedzy

- Rozumie rolę i znaczenie technologii chmury komputerowej oraz zna jej zastosowania w biznesie.
- Zna zasady działania chmur komputerowych w różnych sieciowych systemach operacyjnych.
- Zna techniki i metody wykonania aplikacji chmurowej.

# Efekty uczenia się w zakresie umiejętności

- Potrafi skonfigurować środowisko do pracy w chmurze komputerowej.
- Umie wykorzystać zasoby chmury komputerowej.
- Potrafi zaprojektować i zaimplementować aplikację działającą w chmurze komputerowej.

# Efekty uczenia się w zakresie kompetencji społecznych

- Potrafi myśleć kreatywnie w trakcie analizy i projektowania aplikacji w chmurach komputerowych.

# Metoda weryfikacji

- Egzamin

## Narzędzie weryfikacji

- Test kombinowany: zdania do uzupełnienia, test jednokrotnego i wielokrotnego wyboru

# Stosowane kryteria oceny

- Wpisanie brakujących elementów w zdaniu
- Wybór prawidłowej odpowiedzi

# Próg zaliczeniowy

- 51% łącznej liczby punktów



## Programowanie aplikacji w chmurze obliczeniowej

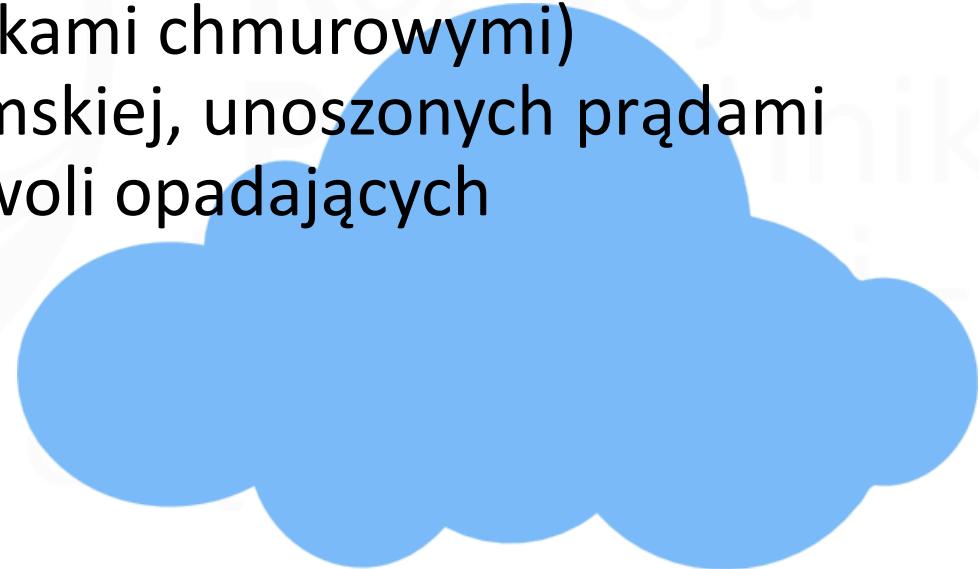
**Pojęcie chmury, rodzaje chmur, koncepcja usług w chmurze**

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni



# Chmura

- Widzialne skupisko mikroskopijnych (o średnicy mniejszej od  $100 \mu\text{m}$ ) cząstek stałych bądź ciekłych (kryształów lodu lub kropel wody, zwanych też cząstkami chmurowymi) w atmosferze ziemskiej, unoszonych prądami powietrza lub powoli opadających



Źródło: Encyklopedia PWN 2010

# Chmura

- Komputerowa/obliczeniowa – encyklopedia PWN, Helionica, Internautica, Wiem.onet – brak definicji
- Chmura obliczeniowa (ang. cloud computing) to model przetwarzania danych oparty na użytkowaniu usług dostarczonych przez usługodawcę - Wikipedia
- Chmura obliczeniowa (cloud computing) to nowoczesny sposób zarządzania danymi – infor.pl
- Chmura obliczeniowa jest to model wykorzystania zasobów informatycznych w postaci usług dostarczanych przez sieć komputerową - Maciej Roszkowski, Zachodniopomorski Uniwersytet Technologiczny, Zeszyty Naukowe nr763, 2013

# Chmura

- Chmura to aplikacje udostępniane przez Internet (cloud), a nie tradycyjne rozwiązania wymagające zakupu i instalacji oprogramowania z pełną licencją - Salesforce.com
- Chmura to setki połączonych komputerów i serwerów z wielordzeniowymi procesorami, dzięki którym można udostępnić moc obliczeniową innym firmom - IBM i Google
- W ramach chmury udostępniane są usługi całej infrastruktury IT, które nie są w pełni wykorzystywane na swoje potrzeby - Amazon.com

# Geneza chmur komputerowych



Superkomputery



Klastry



Systemy gridowe



Chmury komputerowe

Źródła zdjęć: <https://pl.wikipedia.org>

# High Throughput Computing – obliczenia o wielkiej wydajności

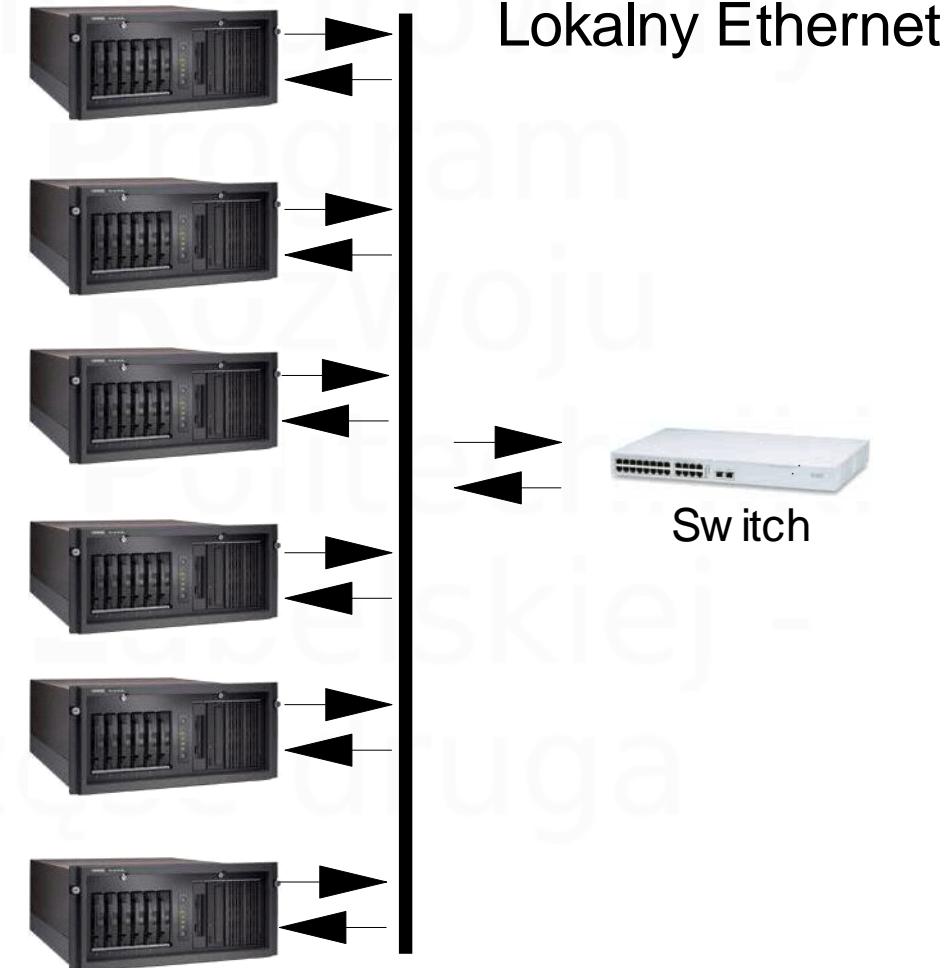
- Dostępne rozwiązania
  - Superkomputery – lata 60 ubiegłego wieku, komputery produkowane przez CDC oraz Cray, koszt kilka milionów dolarów, obecnie produkowane na zamówienie koszt sięga 200 milionów dolarów
- Superkomputery w Polsce:
  - CDC (Control Data Corporation - rok zakupu 1973) - Instytut Badań Jądrowych w Świerku oraz Cyfronet w Krakowie
  - Cray – Cyfronet oraz ICM
  - Zeus – Cyfronet 88 TFLOPS
  - Farma serwerów Nasza Klasa, 46 TFLOPS

# High Throughput Computing – obliczenia o wielkiej wydajności

- Superkomputery w Polsce:
  - farma serwerów QXL Poland (allegro), 105 TFLOPS
  - Galera w TASK w Gdańsku, 65 TFLOPS
  - Nautilus w ICM w Warszawie, 18,5 TFLOPS
    - Komputer ten w listopadzie 2008 roku znalazł się na pierwszym miejscu listy Green500, jako najbardziej energooszczędny superkomputer na świecie z wynikiem 536 MFLOPS/W
  - Blue Gene/Q w ICM w Warszawie, 189 TFLOPS, 14 miejsce na liście Green500 (listopad 2013) z wynikiem 2299,15 MFLOPS/W

# Klastry komputerowe

- klaster wydajnościowy
- klaster niezawodnościowy
- klaster równoważenia



Źródło: opracowanie własne

# Systemy gridowe



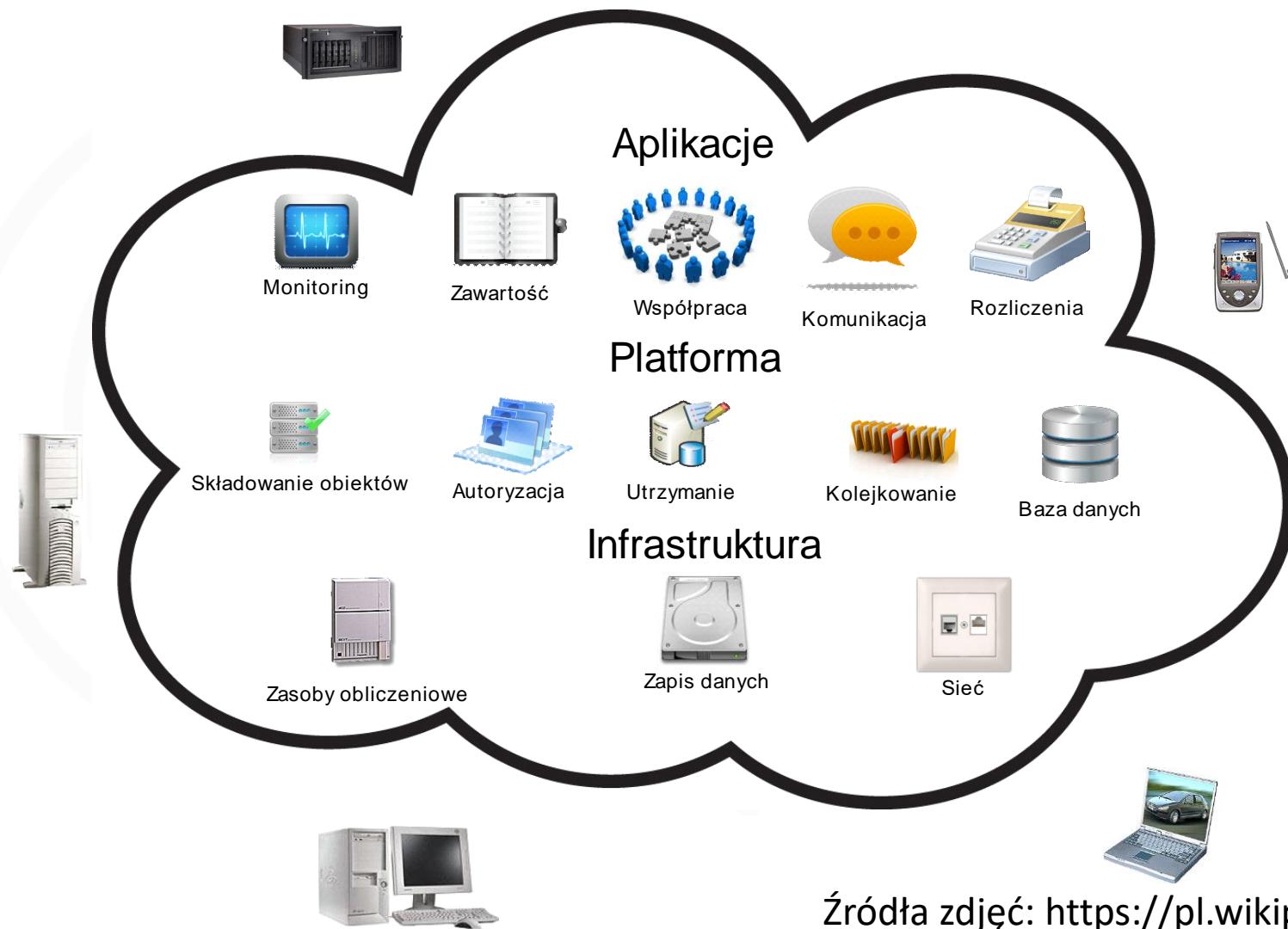
Technika grid computing oparta jest na idei łączenia zasobów danych w metasystemie.

# Chmury komputerowe



źródła zdjęć: <https://pl.wikipedia.org>

# Chmury komputerowe



Źródła zdjęć: <https://pl.wikipedia.org>

# Poziomy chmur komputerowych

- SaaS (ang. SaaS, Software as a Service) – oprogramowanie, jako usługa
  - koncentruje się na oferowaniu internetowych aplikacji, umożliwia wirtualizację niektórych aplikacji biurowych
- PaaS (ang. PaaS, Platform as a Service) - platforma jako usługa
  - wirtualne stanowiska pracy dla programistów, gdzie system operacyjny i narzędzia programistyczne znajdują się w „chmurze”, przez co użytkownik nie musi sam ponosić kosztów ich pozyskania oraz utrzymania

# Poziomy chmur komputerowych

- IaaS (ang. IaaS, Infrastructure as a Service) - infrastruktura jako usługa
  - wirtualizacja dostępu do infrastruktury, uzyskanie przez użytkownika dostępu do skalowalnej usługi, dostosowującej się do jego potrzeb
- CaaS (ang. Communications as a Service) - komunikacja jako usługa
  - usługodawca zapewnia platformę pod telekomunikacyjne środowisko pracy
- iPaaS (ang. Integration Platform as a Service ) - platforma integracyjna jako usługa
  - platforma zapewniająca integrację pomiędzy różnymi usługami w chmurze



Programowanie aplikacji w chmurze obliczeniowej

**Wykorzystanie technik wirtualizacji w chmurach  
komputerowych**

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni



**Fundusze  
Europejskie**  
Wiedza Edukacja Rozwój

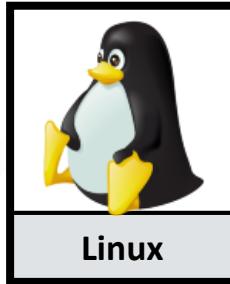


**Rzeczpospolita  
Polska**

**Unia Europejska**  
Europejski Fundusz Społeczny



# Technologia wirtualizacji



Linux



Linux dev



XP



Win7



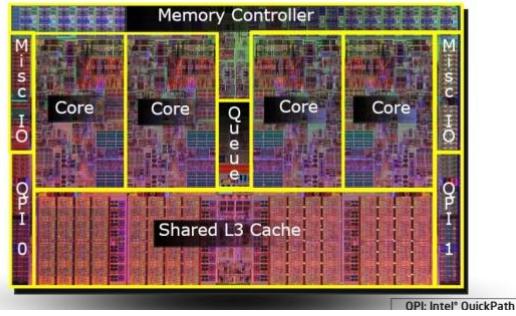
MacOS

Virtual Machine Monitor

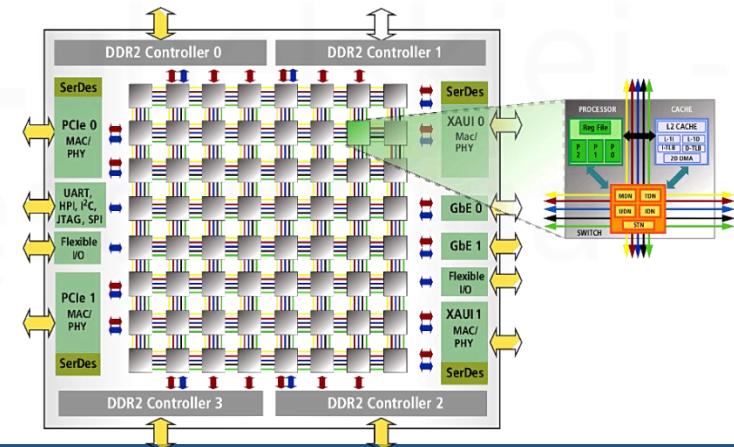
Hardware

4 rdzeniowy Intel

The First Nehalem Processor

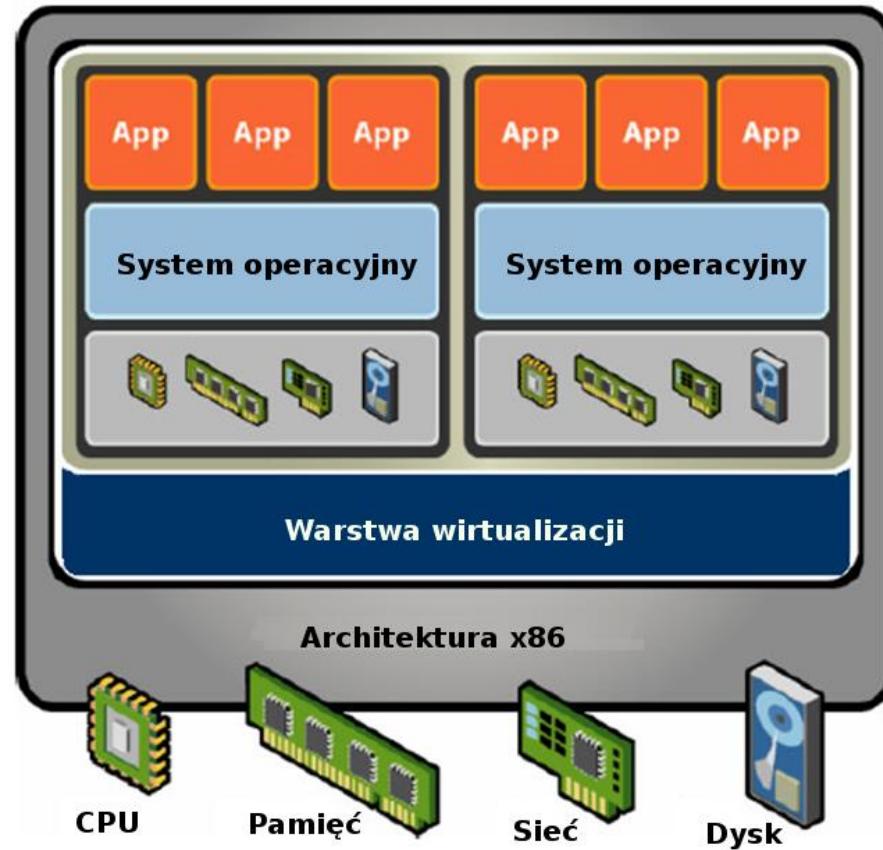


100 rdzeniowa Tilera



Źródło: <https://www.intel.com>

# Technologia wirtualizacji



Źródło: <https://www.vmware.com>

- Pełna wirtualizacja
  - Wiele systemów gości, kompletna separacja zasobów
  - Duży narzut, często kosztowne rozwiązania
- Parawirtualizacja
  - Wiele systemów gości, mały narzut
  - Wymaga wsparcia sprzętowego lub zmodyfikowanego systemu operacyjnego gospodarza

# Izolacja maszyn wirtualnych

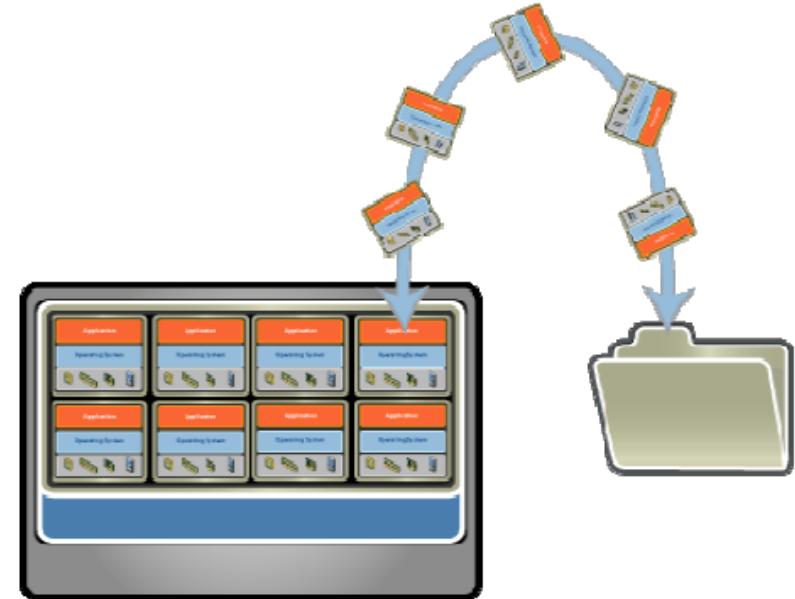
- Bezpieczny multiplexing
  - Uruchamianie wielu VM na pojedynczymホście
  - Sprzętowy procesor izoluje VM
- Bezpieczeństwo
  - Błędy programowe, awarie, wirusy nie wpływają na inne VM
- Izolacja wydajności
  - Zasoby systemowe spartycjonowane
  - Np. możliwości definicji zasobów w VMware



Źródło: <https://www.vmware.com>

# Enkapsulacja maszyn wirtualnych

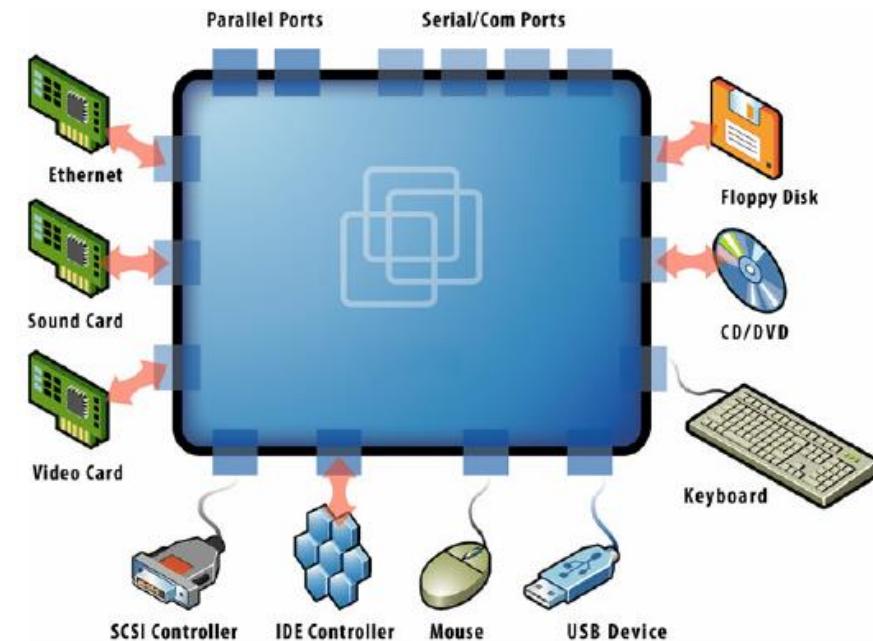
- Maszyna to Plik
  - system, aplikacje, dane
  - pamięć i stan urządzenia
- Migawki i klonowanie
  - możliwość zatrzymania maszyny w czasie
  - szybkie kopie zapasowe, klonowanie itp.
- Łatwość dystrybucji
  - prekonfigurowane aplikacje
  - wirtualne przyrządy



Źródło: <https://www.vmware.com>

# Kompatybilność maszyn wirtualnych

- Niezależne sprzętowo
  - sprzęt ukryty za warstwą wirtualizacji
  - zwirtualizowany sprzęt dostępny dla VM
- Utwórz raz, uruchom gdziekolwiek
  - brak zmian w konfiguracji
  - migracja maszyn między hostami
- Zgodność maszyn VM
  - możliwość uruchomienia starych systemów operacyjnych



Źródło: <https://www.vmware.com>

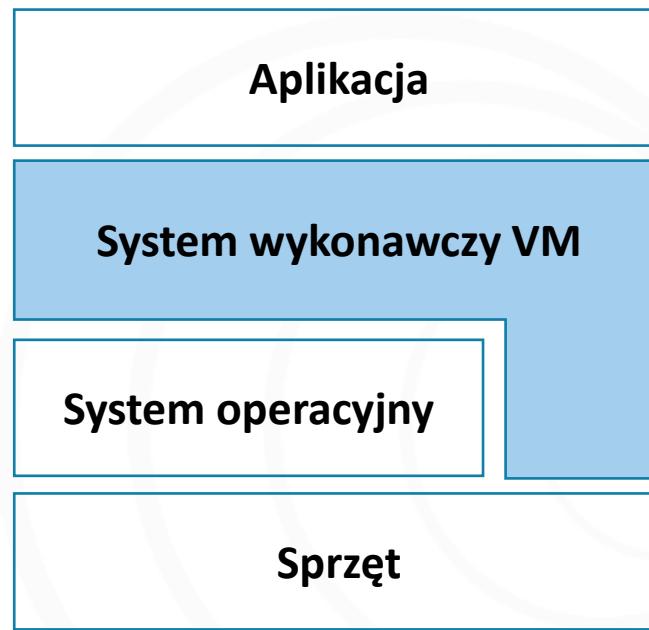
# Powszechnie wykorzystanie wirtualizacji

- Uruchamianie starego oprogramowania na niezgodnej z nim platformie sprzętowej
- Uruchamianie wielu systemów operacyjnych na jednej platformie sprzętowej
- Tworzenie zarządzalnej ścieżki aktualizacji
- Dynamiczne zarządzanie awariami (spodziewanymi i niespodziewanymi)
- Zastosowanie w technologiach sieci gridowych i chmurowych

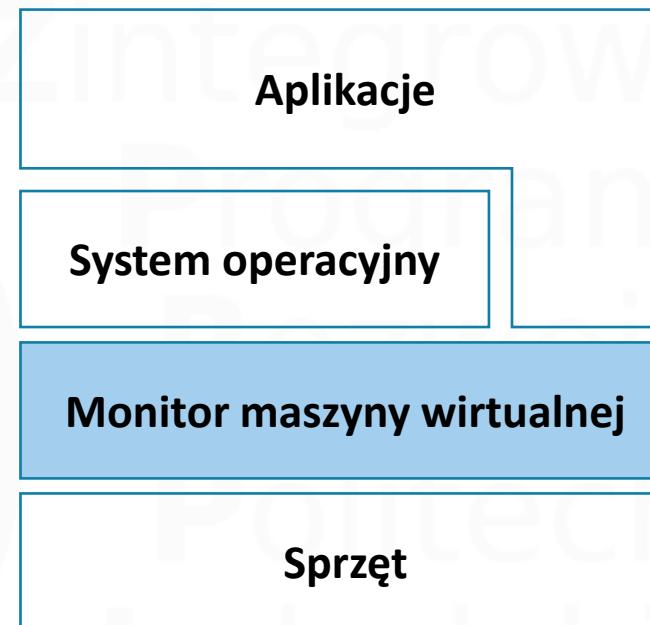
# Dynamiczne centra danych

- Wirtualizacja pomaga złamać model „jeden serwis na serwer”
- Konsolidacja wielu serwisów do kilku maszyn pozwala na redukcję kosztów (wykorzystanie wolnych zasobów)
- Jeżeli popyt na poszczególne usługi wzrasta, to można uruchomić więcej maszyn wirtualnych
- Można zbudować centrum danych z mniejszą liczbą całkowitych zasobów, ponieważ wykorzystywane są zasoby w miarę potrzeb, a nie są poświęcane dla pojedynczych usług

# Wirtualizacja a maszyna wirtualna



A)



B)

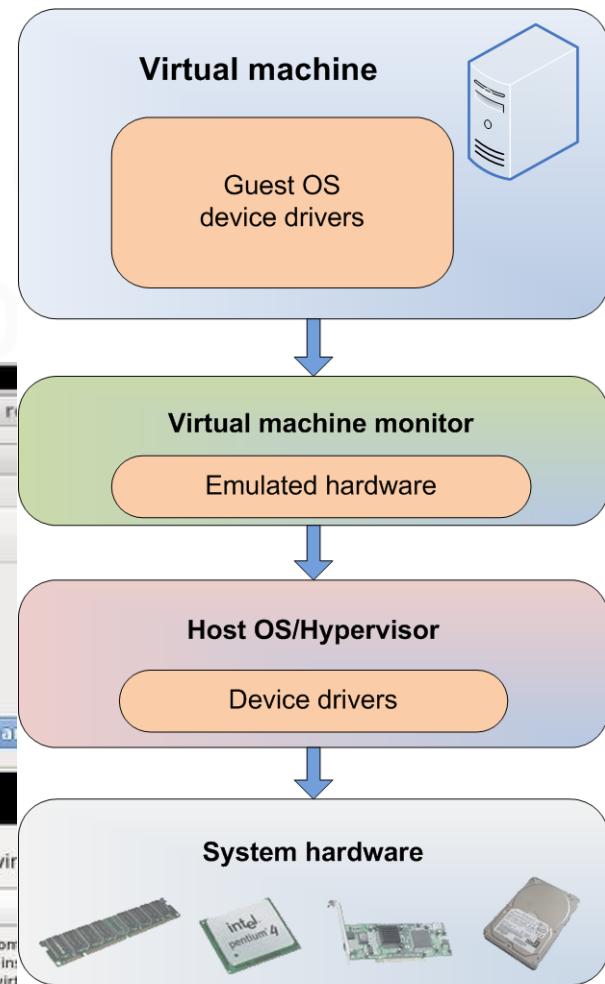
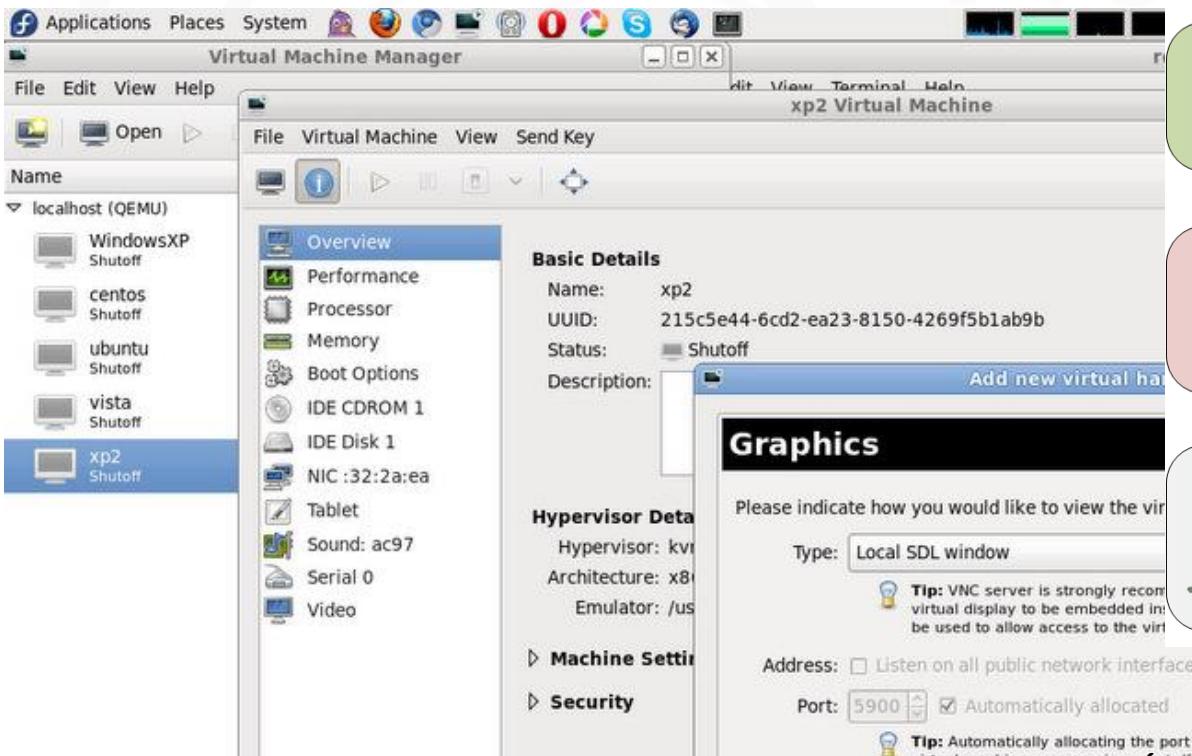
- A) VM procesu – program jest kompilowany do kodu pośredniego, który następnie jest wykonywany przez system wykonawczy (np. Java VM)
- B) VM monitor – oddzielna warstwa programowa działająca jak sprzętowa (np. VMware, VirtualBox)

# Trzy rozwiązania wirtualizacji

- Pełna wirtualizacja
- Parawirtualizacja
- Sprzętowo-wspomagana wirtualizacja

# Pełna wirtualizacja

- Wszystko jest wirtualizowane
- Pełna emulacja sprzętu
- Emulacja=opóźnienia

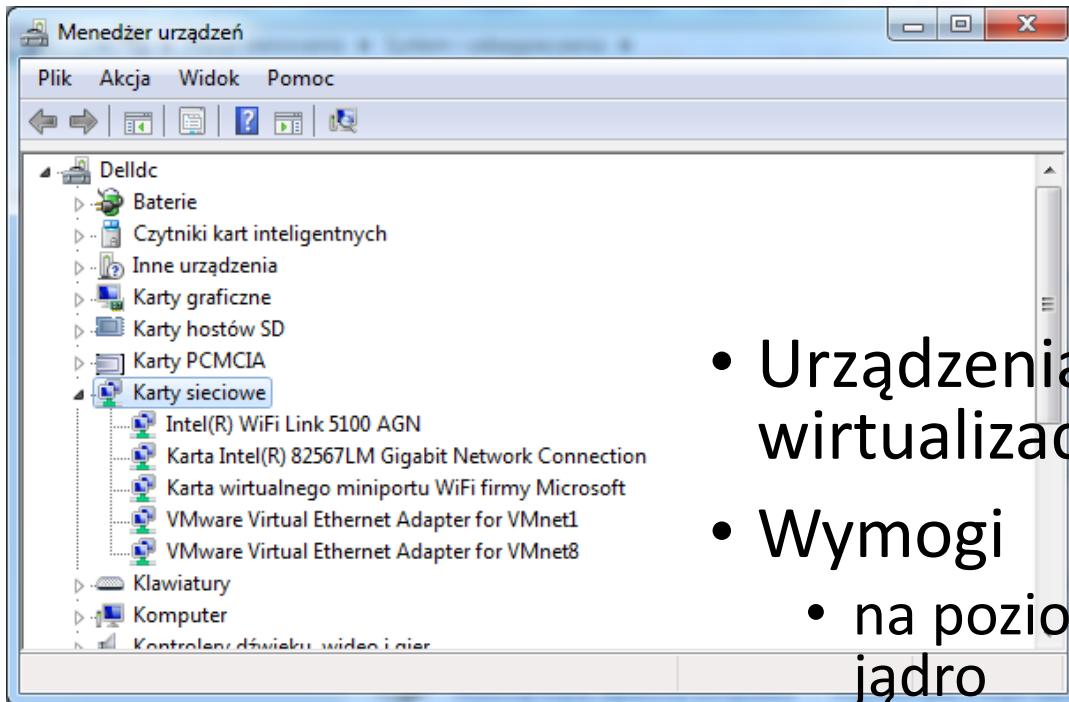


Źródło: <https://www.vmware.com>

# Pełna wirtualizacja – za i przeciw

- Za
  - Odporność na awarie
  - Rozprzestrzenianie gotowych maszyn wirtualnych
  - Oprogramowanie na inne architektury sprzętowe
- Przeciw - OPÓŹNIENIA
  - Pamięć RAM wydajność zredukowana do 25% - 75%
  - Obniżenie wydajności operacji dyskowych w granicach 5% do 20%
  - Wydajność sieci do 10%
  - Uprzywilejowane instrukcje CPU 1-7%

# Parawirtualizacja



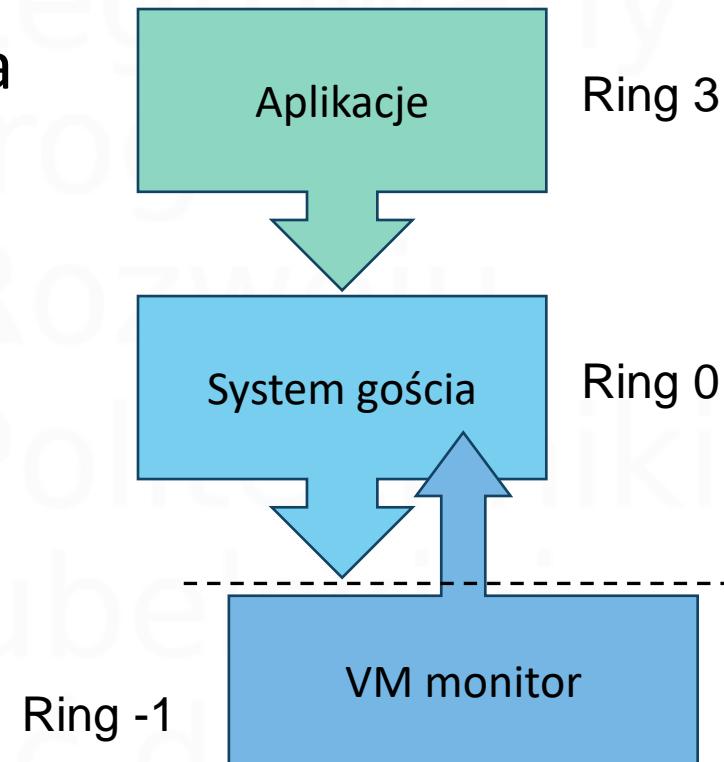
- Urządzenia są świadome wirtualizacji
- Wymogi
  - na poziomie OS – zmodyfikowane jądro
  - na poziomie urządzeń – odpowiednie serowniki wspierające parawirtualizację

# Parawirtualizacja

- Za
  - Prędkość
- Przeciw
  - Wymaga specjalnie zmodyfikowanego systemu-goszcia, co wyklucza możliwość bezpośredniej pracy systemów hermetycznych i starszych w środowiskach parawirtualnych

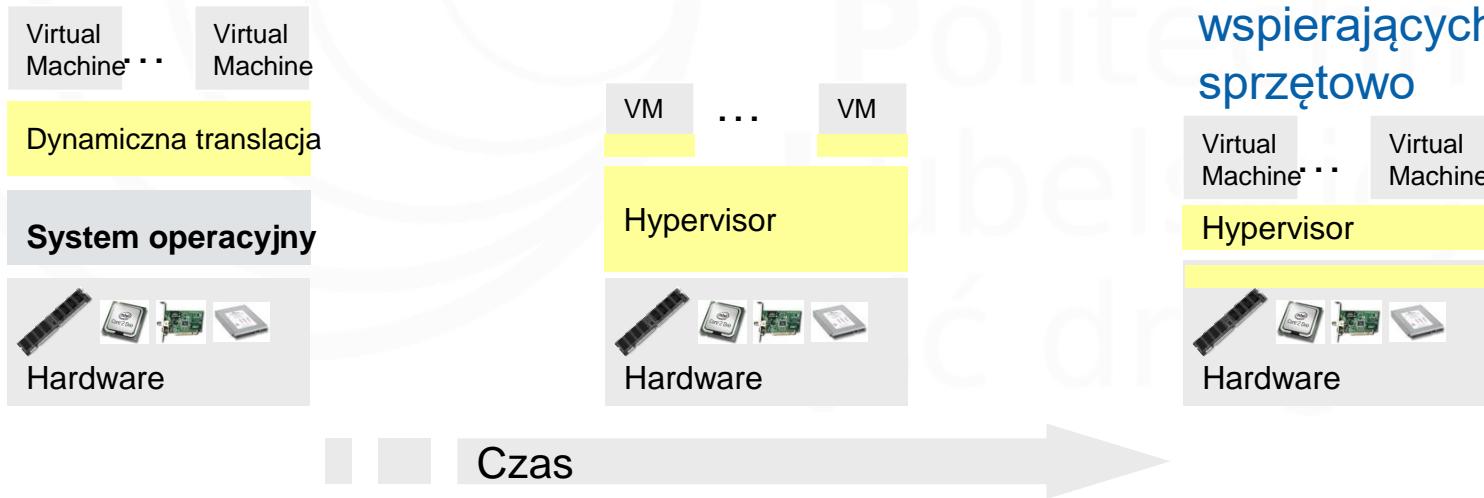
# Wirtualizacja wspomagana sprzętowo

- Istnieje sprzętowe wsparcie dla wirtualizacji
- Hypervisor i VMM są umieszczone w uprzywilejowanym Ring -1 (firmware)
- Usuwa wąskie gardło emulacji procesora
- Wirtualizacja pamięci w rozwiązaniach QuadCore AMD i Intela



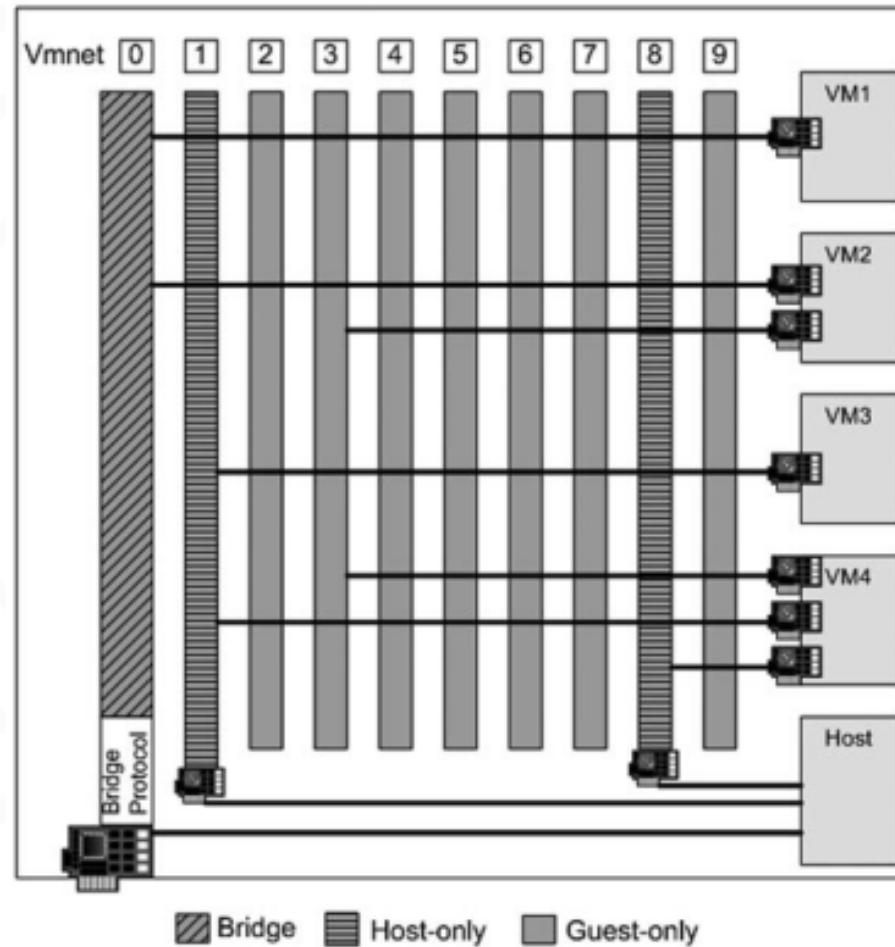
# Ewolucja rozwiązań programowych\*

- 1<sup>st</sup> Generacja: Pełna wirtualizacja (przepisywanie binariów)
  - Programowa
  - VMware i Microsoft
- 2<sup>nd</sup> Generacja: Parawirtualizacja
  - Wirtualizacja kooperatywna
  - Modyfikowany gość
  - VMware, Xen
- 3<sup>rd</sup> Generacja: Krzemowa (wpierana sprzętowo)
  - Niemodyfikowany gość
  - VMware i Xen na platformach wspierających sprzętowo



Źródło: \*Materiały prezentacja firmy Intel®

# Wirtualne sieci VMware



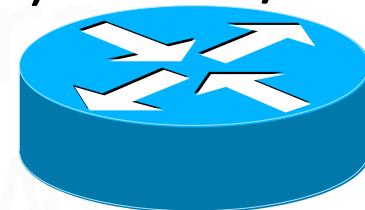
Źródło: <https://www.vmware.com>

# Wirtualne interfejsy sieciowe

- VMnet0 - jest jedną siecią, która jest bezpośrednio podłączone do sieci fizycznej poprzez mostek.
- VMnet1 and VMnet8 – w interfejsach VMnet1 i VMnet8 system posiada wirtualną kartę sieciową, a więc może komunikować się z maszynami wirtualnymi i posiada również funkcję bramy w sieci NAT.
- Maszyny wirtualne VM1 do VM4 wyposażone są w różną liczbę wirtualnych kart sieciowych, które z kolei są przypisane do różnych sieci VMnet.

# Dostępne konfiguracje VMware

- Współdzielenie folderów (Shared folders) brama/router:

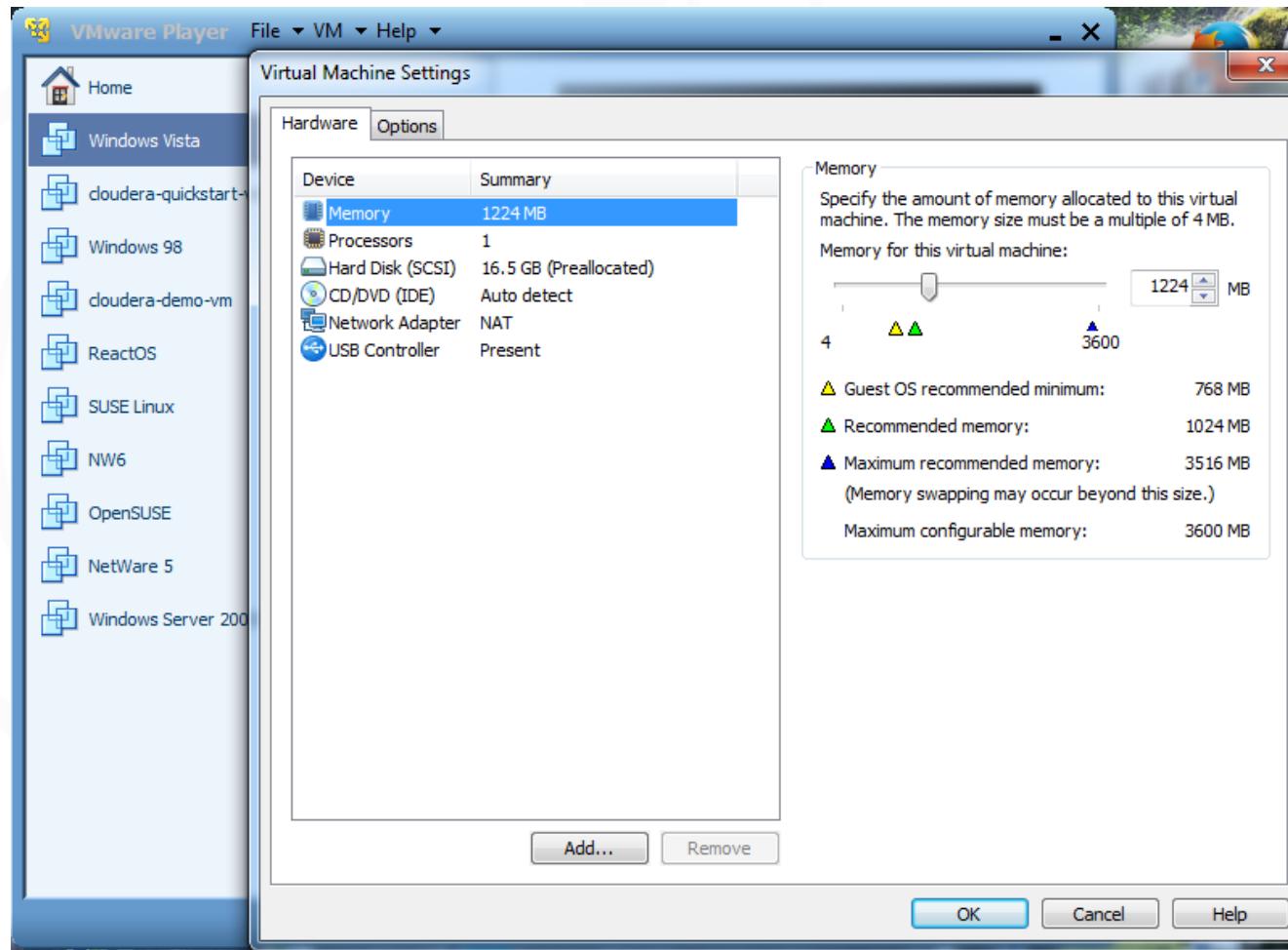


- NAT router z przełącznikiem LAN
- Mostek – bridge bez wewnętrznego przełączania sieci LAN
- Router „Host-only” (tylko host) z przełączaniem sieci LAN
- Sieć wewnętrzna (Internal Network) LAN z opcjonalnymi serwisami DHCP

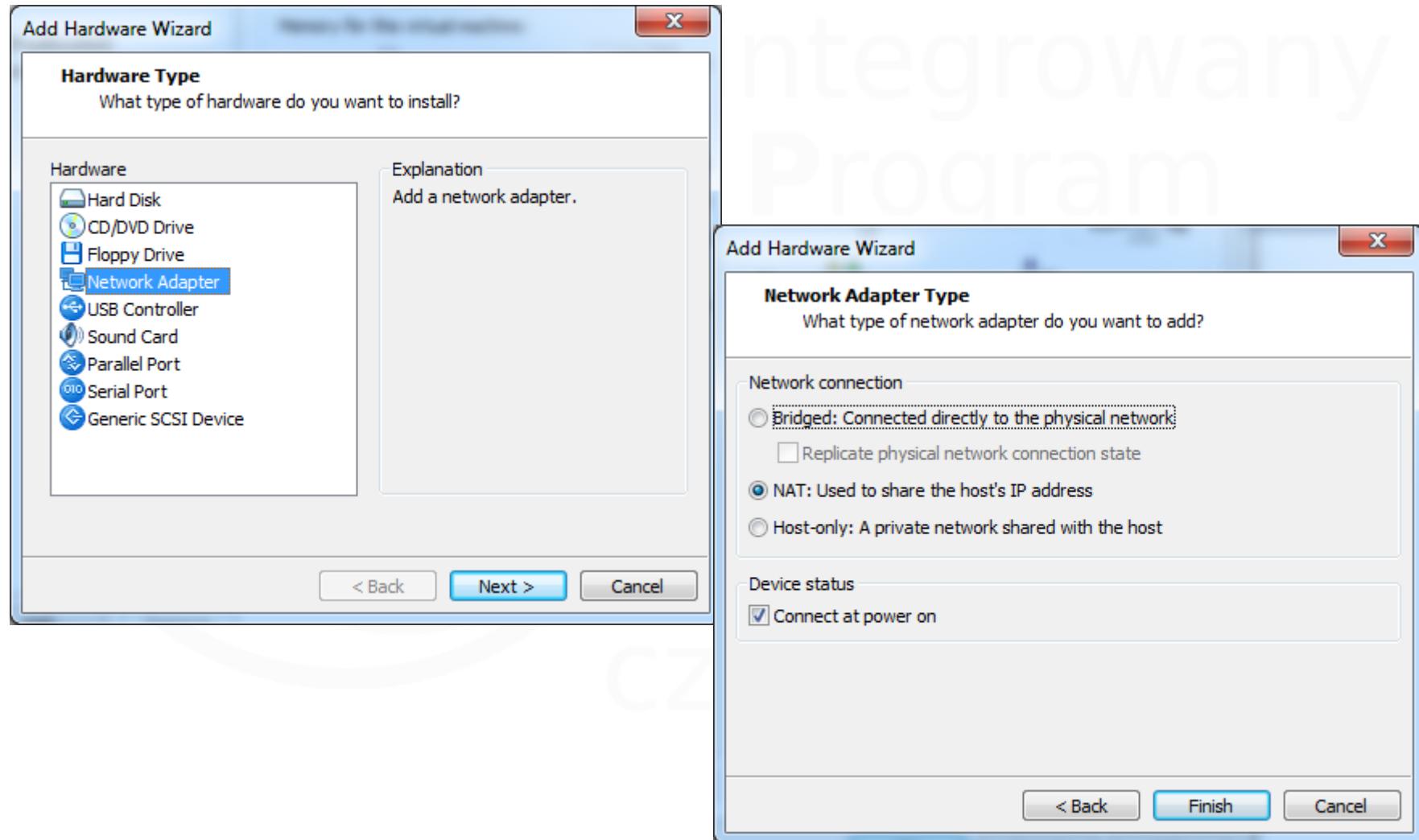
# Rodzaje wirtualnych połączeń sieciowych

- Host-Only: do VM zostanie przydzielony jeden adres IP, ale dostępny tylko w oknie gościa. Żadne inne komputery nie mają do niego dostępu.
- NAT: Podobnie jak w sieci domowej z routerem bezprzewodowym, do VM zostanie przydzielony adres IP w osobnej podsieci, jak 192.168.6.1 jest komputerem hostem i VM jest 192.168.6.3, wówczas maszyna wirtualna może uzyskać dostęp do sieci na zewnątrz, jak host, ale dostępu z zewnątrz do VM bezpośrednio jest chroniony.
- Zmostkowane: jeżeli VM będzie w tej samej sieci co host, i jeśli IP hosta to 172.16.120.45 wówczas VM będzie np. 172.16.120.50. Wówczas VM jest dostępny dla wszystkich komputerów w sieci gospodarza.

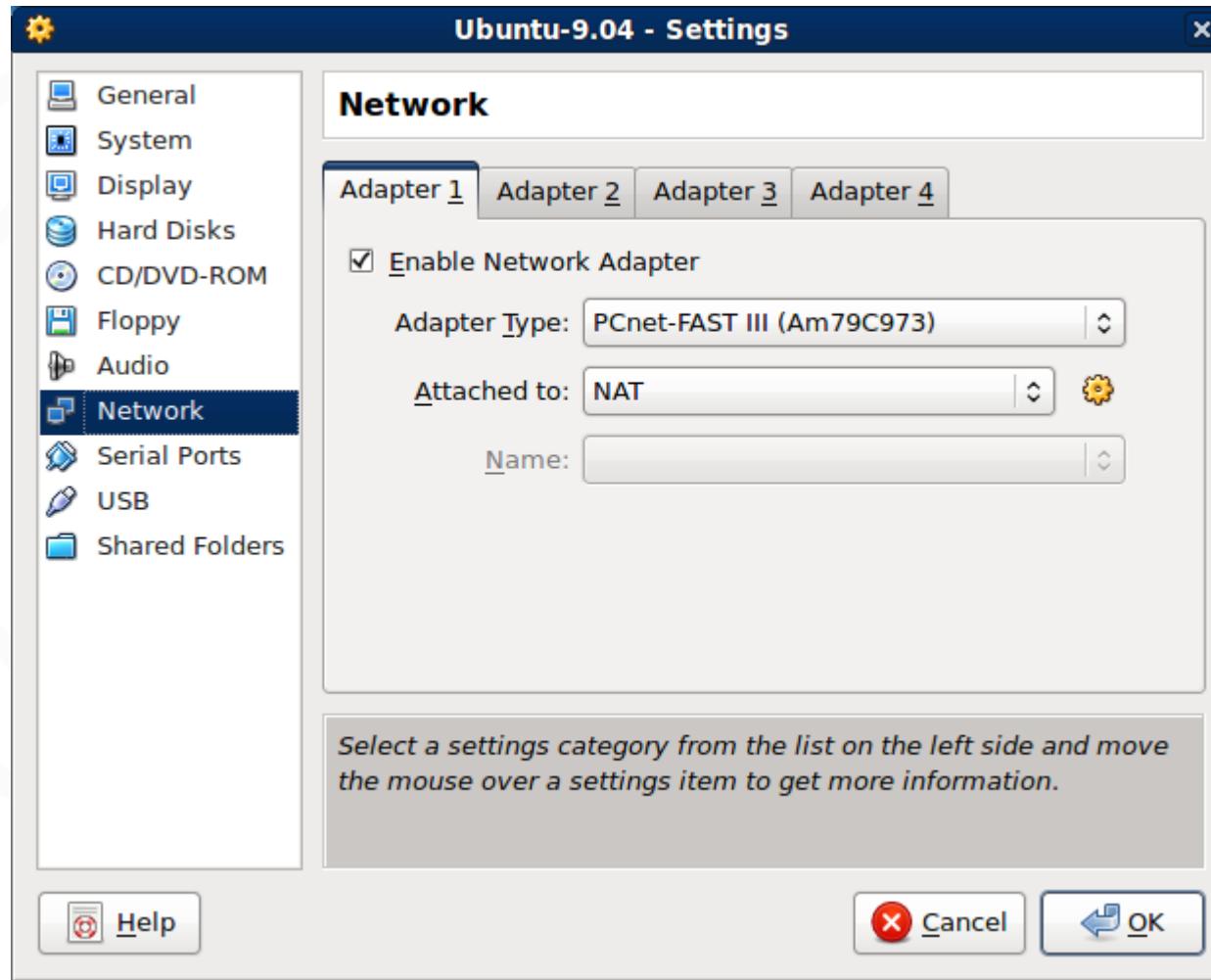
# VMware konfiguracja sprzętowa



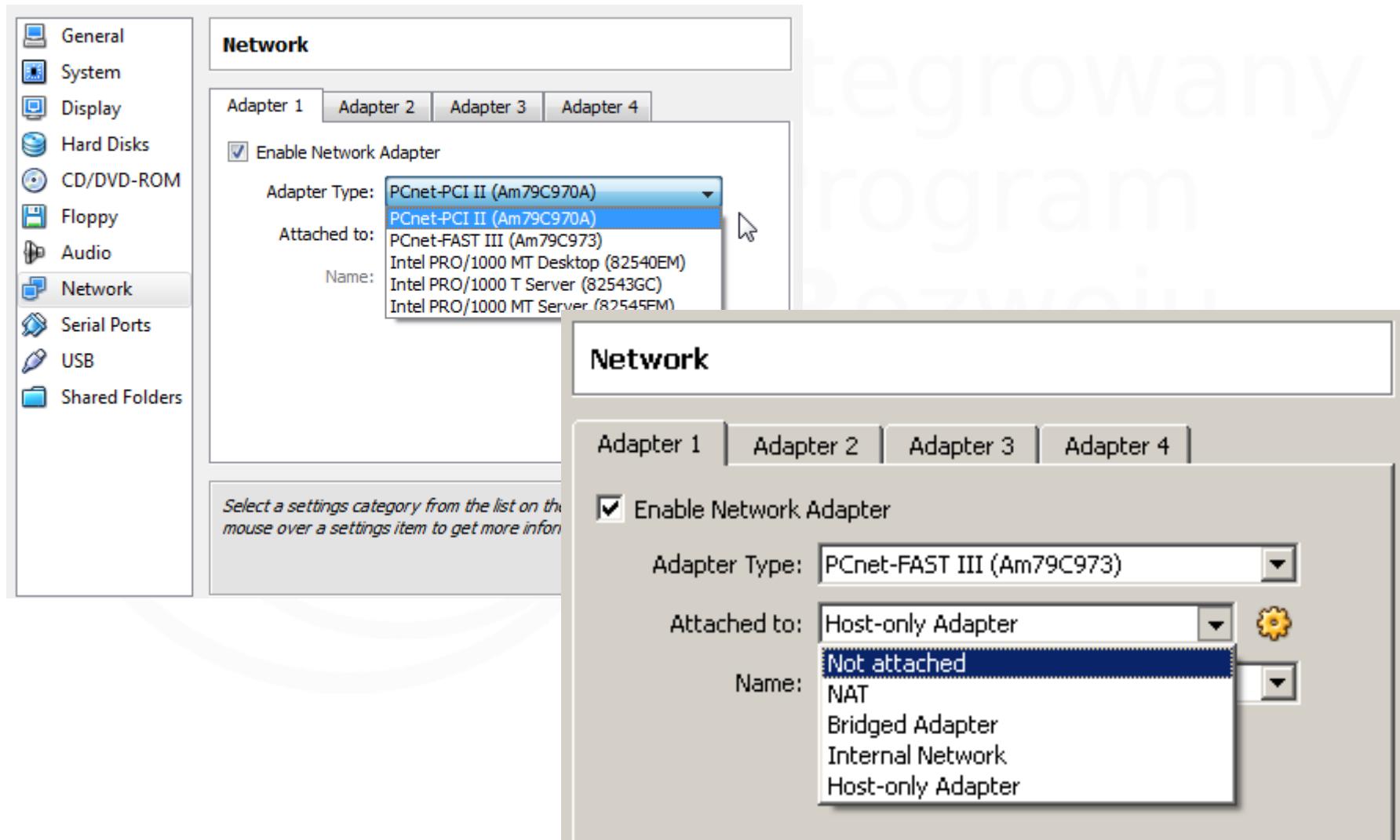
# VMware konfiguracja sprzętowa



# Virtual Box – interfejsy sieciowe



# Virtual Box – interfejsy sieciowe





## Programowanie aplikacji w chmurze obliczeniowej

## Obsługa pamięci masowych w chmurze komputerowej

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni



**Fundusze  
Europejskie**  
Wiedza Edukacja Rozwój



**Rzeczpospolita  
Polska**

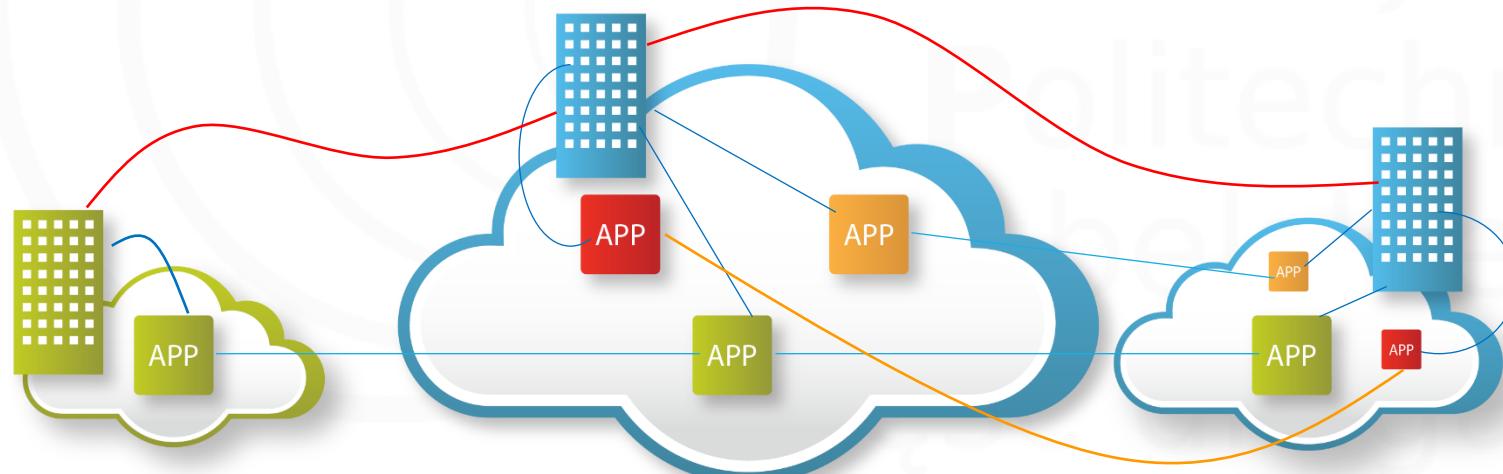
**Unia Europejska**  
Europejski Fundusz Społeczny



# Wirtualizacja – konieczność kompatybilności

**Dowolna lokalizacja**

**Dowolna aplikacja/Dane**



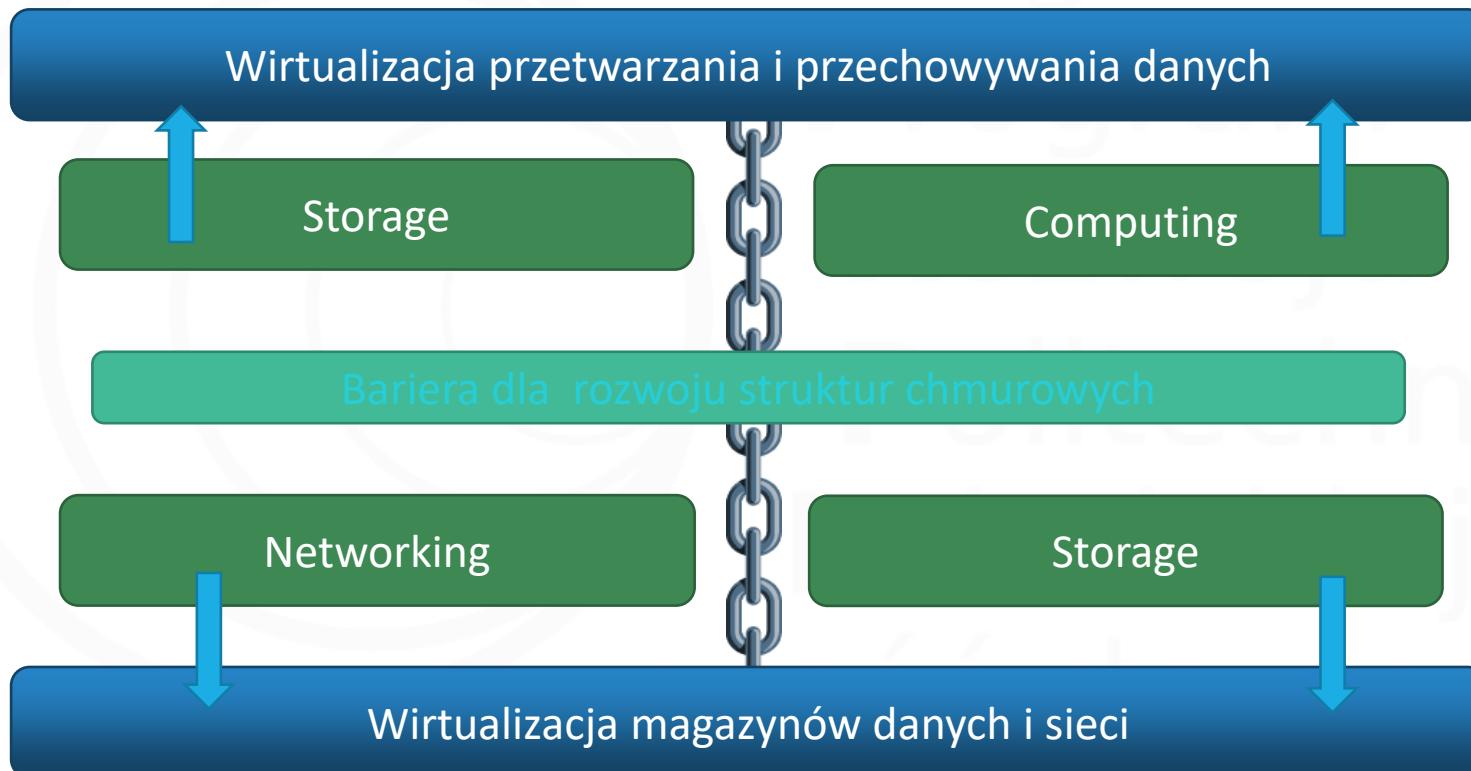
Odbiorca usług/danych      Internet/chmura komputerowa      Dostawca usług/danych

dane-dane

aplikacja-dane

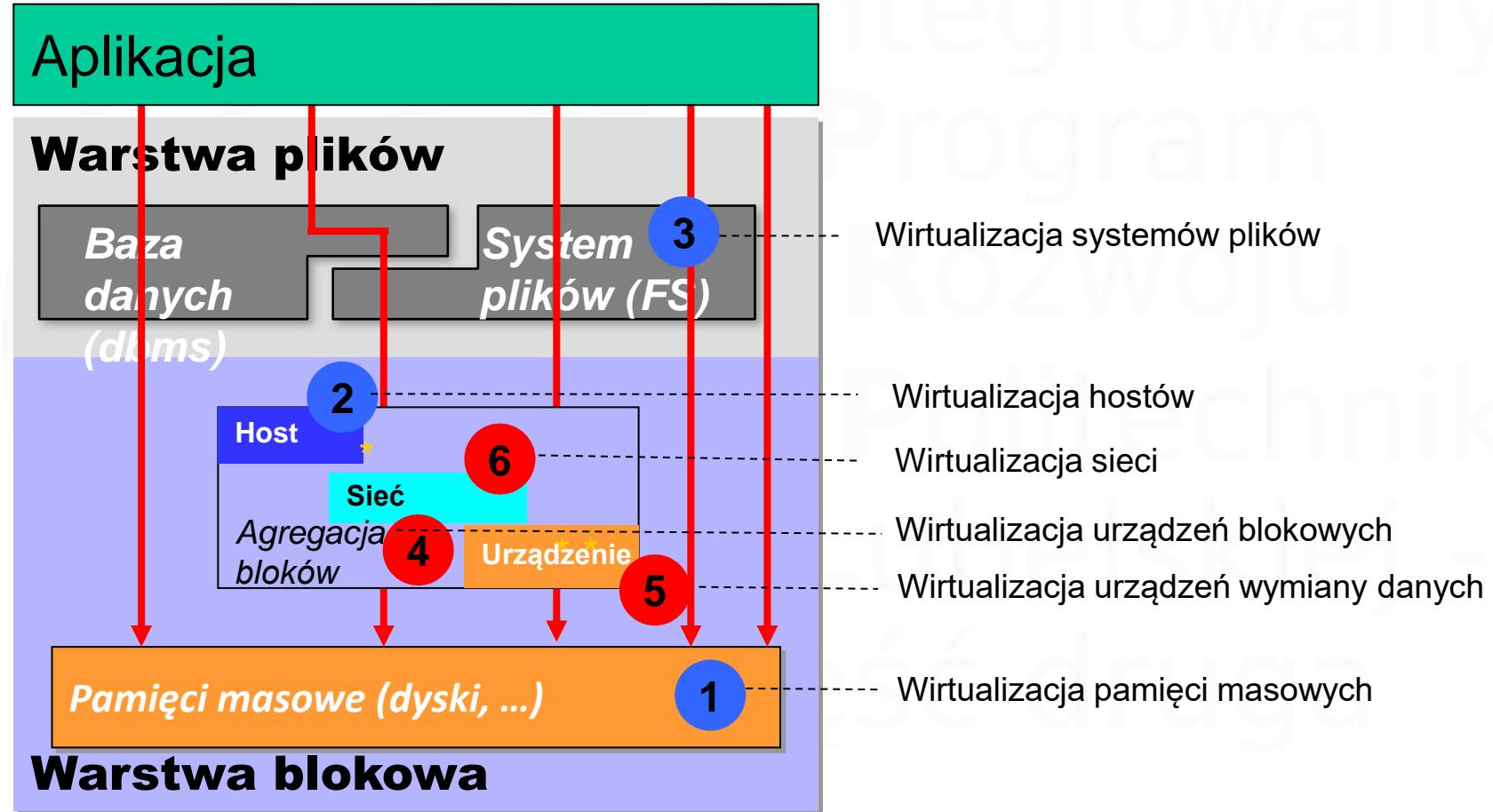
aplikacja-aplikacja

# Wirtualizacja – bariery do pokonania



Źródło: opracowanie własne

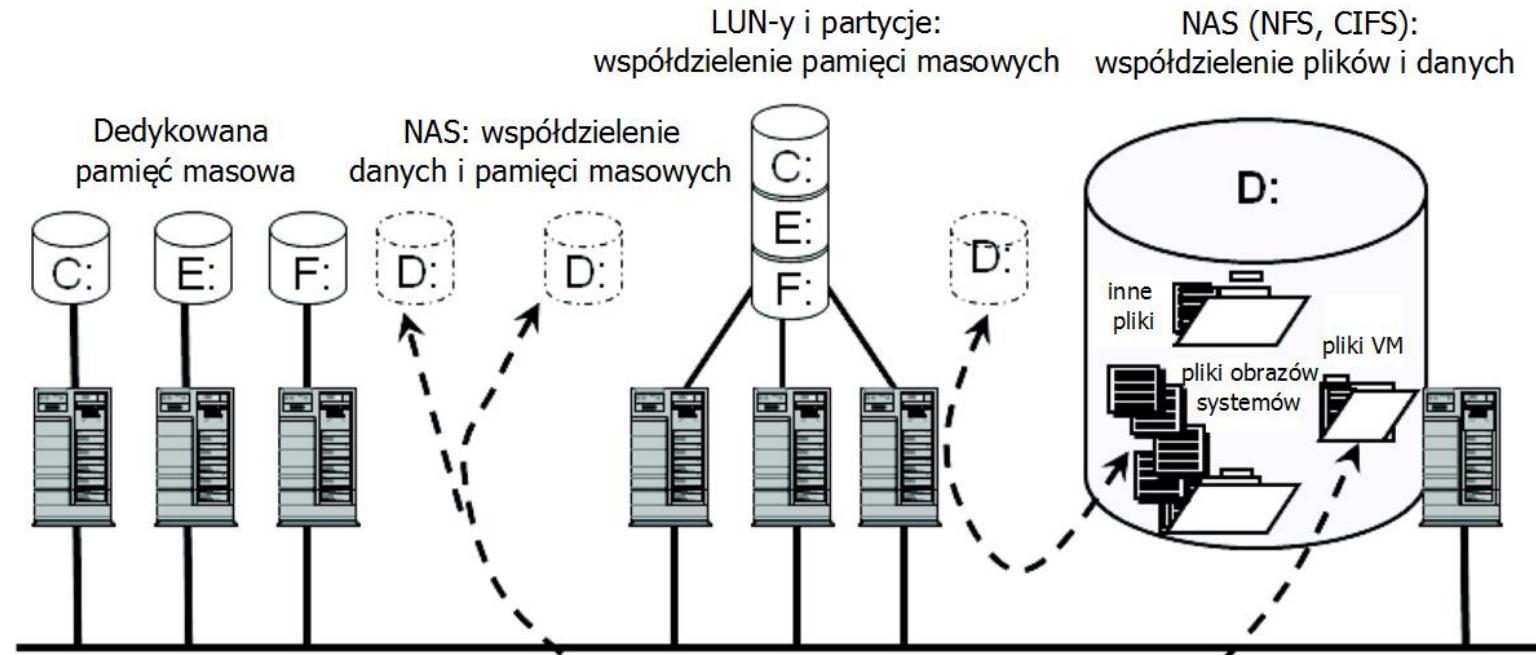
# Wirtualizacja – bardziej szczegółowe spojrzenie



# Współdzielenie danych i pamięci masowych (1)

- W przypadku współdzielenia pamięci masowych (ang. shared storage), poszczególne serwery otrzymują dostęp do części pamięci, typowo zorganizowanej jako partycja, dysk lub wolumen logiczny lub LUN (ang. Logical Unit Number).
- Współdzielenie danych (ang. shared data) związane jest z odczytem lub zapisem danych przez wiele serwerów z pojedynczego pliku. Realizacja tego rodzaju współdzielenia odbywa się za pośrednictwem oprogramowania i dedykowanych protokołów, np. NFS (ang. Network File System) czy CIFS (ang. Common Internet File System).

# Współdzielenie danych i pamięci masowych (2)

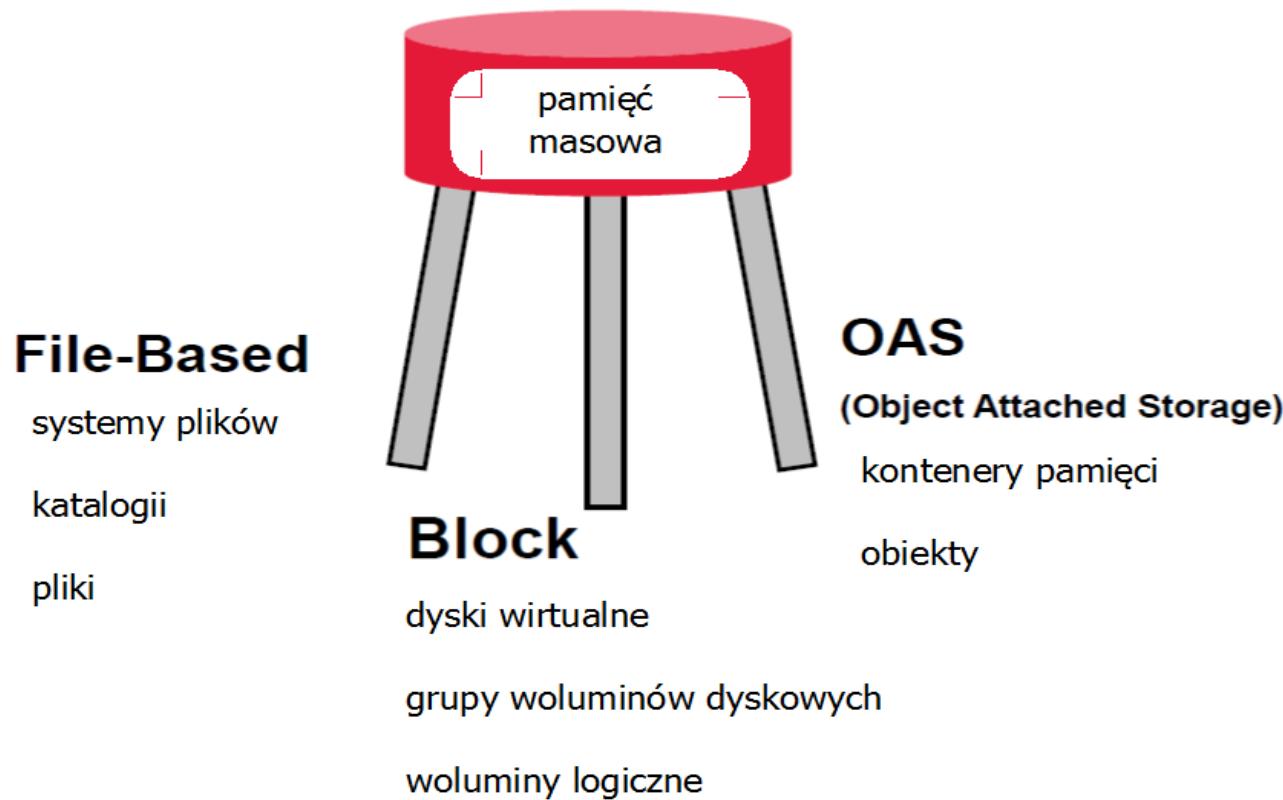


Źródło: opracowanie własne

# Metody dostępu do danych i pamięci masowych (1)

- Najbardziej ogólna klasyfikacja dostępu (lokalnego, zdalnego czy za pośrednictwem chmury) do danych i pamięci masowych obejmuje dostęp:
  - za pośrednictwem API
  - blokowy (ang. block-based access) – structured data
  - plikowy (ang. file-based access) - unstructured data
  - obiektowy (ang. object-based access) – structured data

# Metody dostępu do danych i pamięci masowych (2)



# Dostęp blokowy

- Dostęp blokowy jest najniższym poziomem dostępu do danych i tym samym tworzy fundament dla wszystkich pozostałych tj, jest podstawą systemów dostępu do pamięci masowych dla struktur chmurowych jak i sieciowych pamięci masowych.
- W przypadku dostępu za pośrednictwem systemu plików, baz danych, systemów zarządzania dokumentami itp. szczegóły dostępu blokowego są ukrywane przez określony rodzaj abstrakcji. Abstrakcje te mogą przyjmować zróżnicowane formy, zależne od miejsca ich implementacji, np. systemy dyskowe ze wsparciem LVM, kontrolery RAID itd.

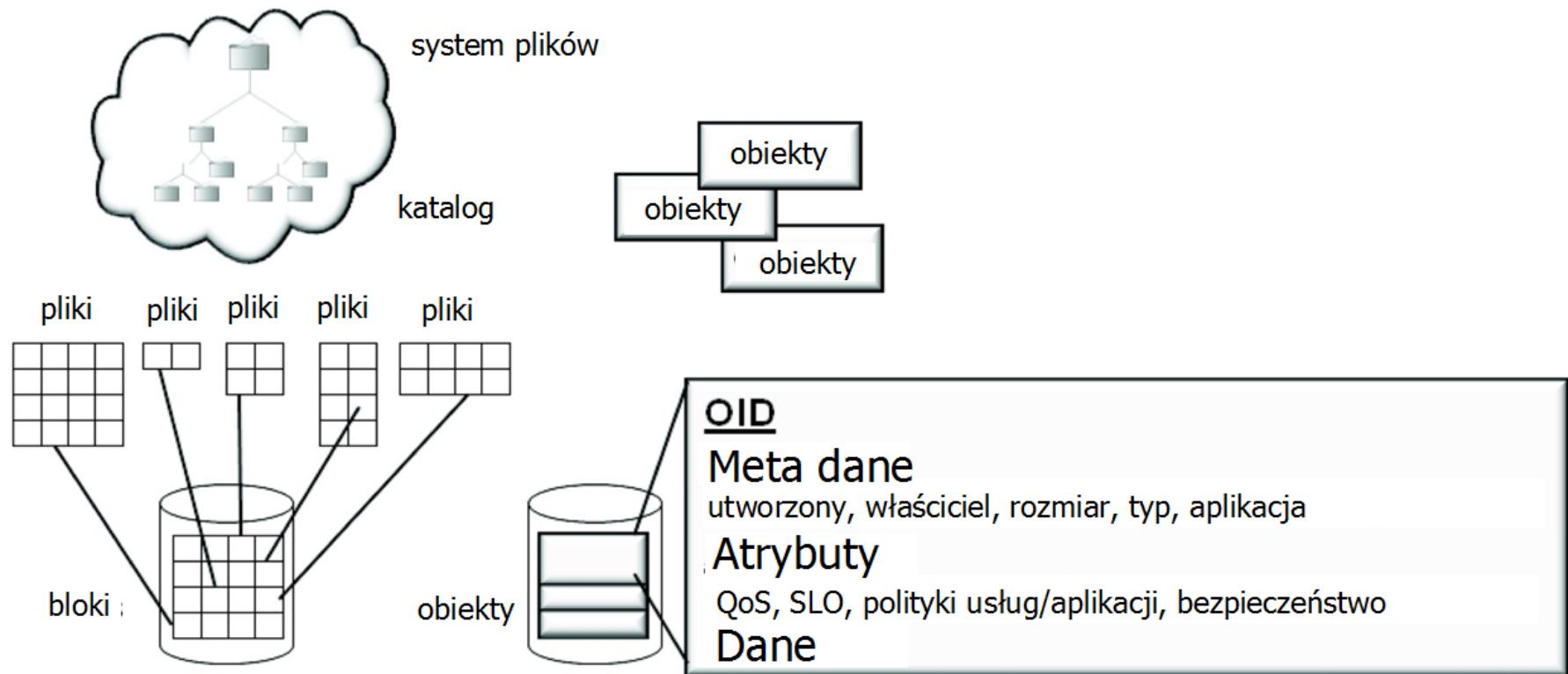
# Dostęp plikowy (1)

- Dostęp plikowy jest obecnie najpopularniejszym sposobem dostępu. Dominuje w sieciowych systemach pamięci ale jest równie popularny w środowiskach zwirtualizowanych. W systemach chmurowych jest obecnie konkurentem dostępu obiektowego.
- Dostęp do danych tego typu wykorzystuje abstrakcję dla ukrycia dostępu blokowego w postaci nazwy pliku. Odnosi się zatem do danych bez predefiniowanej struktury.
- Operacje na danych wykorzystują określony typ semantyki poleceń, np. open, write itd. Dane z pliku pobierane są w blokach o określonym rozmiarze.

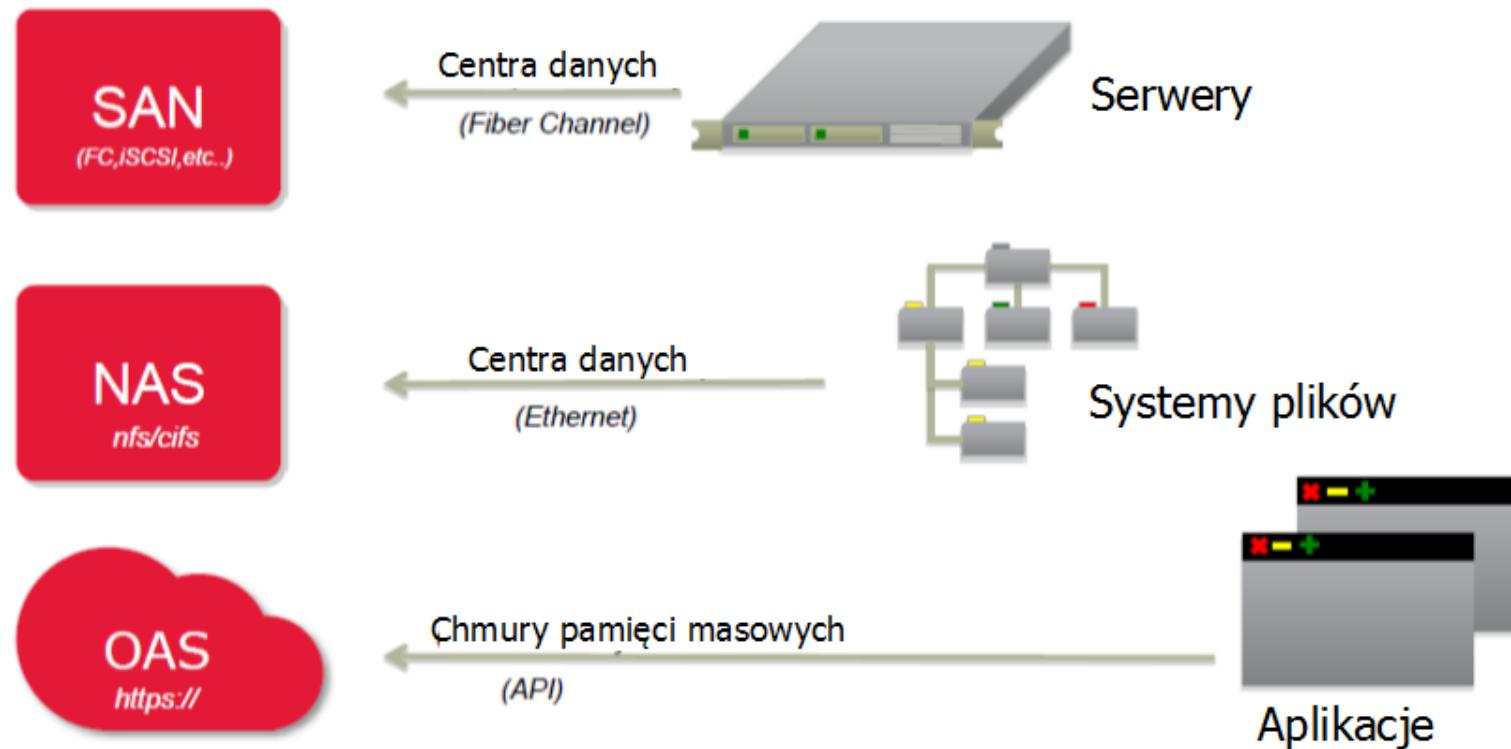
# Dostęp obiektowy (1)

- Dostęp obiektowy dotyczy pamięci CAS (ang. content-addressable storage) czyli pamięci masowych o dostępie zbudowanych na bazie dwóch poprzednio omówionych rozwiązań. W tym mechanizmie dostępu dane są połączone z metadanymi (danymi opisującymi dane). Sposób i miejsce zapisu jest uzgadniane pomiędzy określoną aplikacją (procesem) a systemem plików a dalej, za jego pośrednictwem, z mechanizmami dostępu blokowego.
- Przykład meta danych to rozmiar pliku, informacja kto utworzył plik i ewentualnie dla kogo, atrybuty odczytu i zapisu, dane zabezpieczeń, typ danych ....

# Dostęp obiektowy (2)



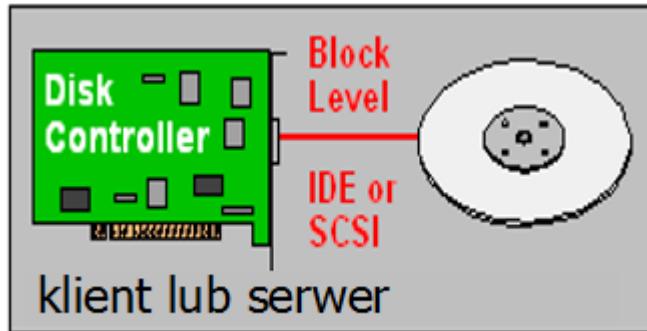
# Inne spojrzenie na metody dostępu



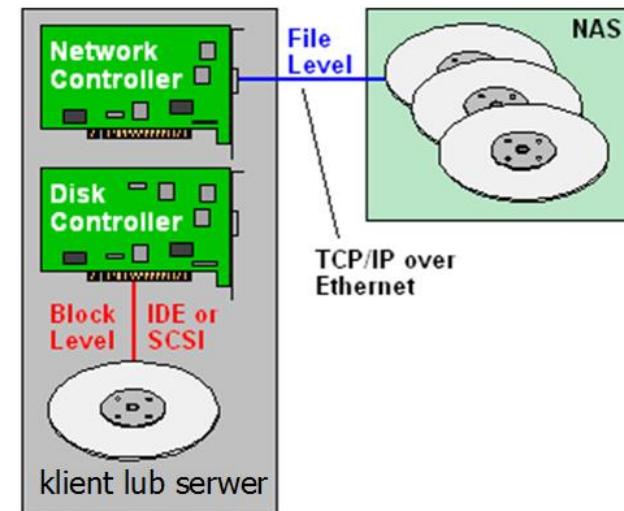
# Trzy podstawowe struktury pamięci masowych (1)

- Direct access storage (DAS)
- Network attached storage (NAS)
- Storage area network (SAN)

## Direct Attached Storage (DAS)



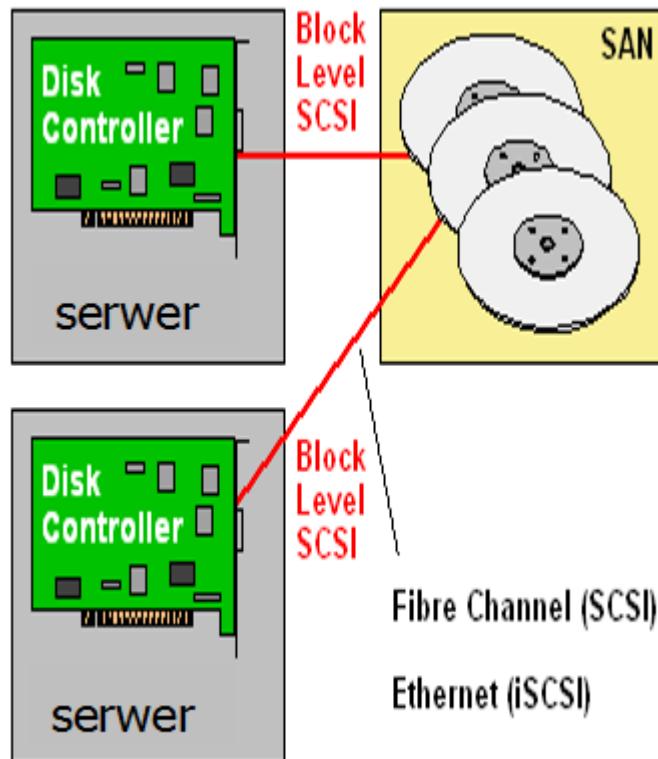
## Network Attached Storage (NAS)



Źródło: Computer Desktop Encyclopedia

# Trzy podstawowe struktury pamięci masowych (2)

## Storage Area Network (SAN)

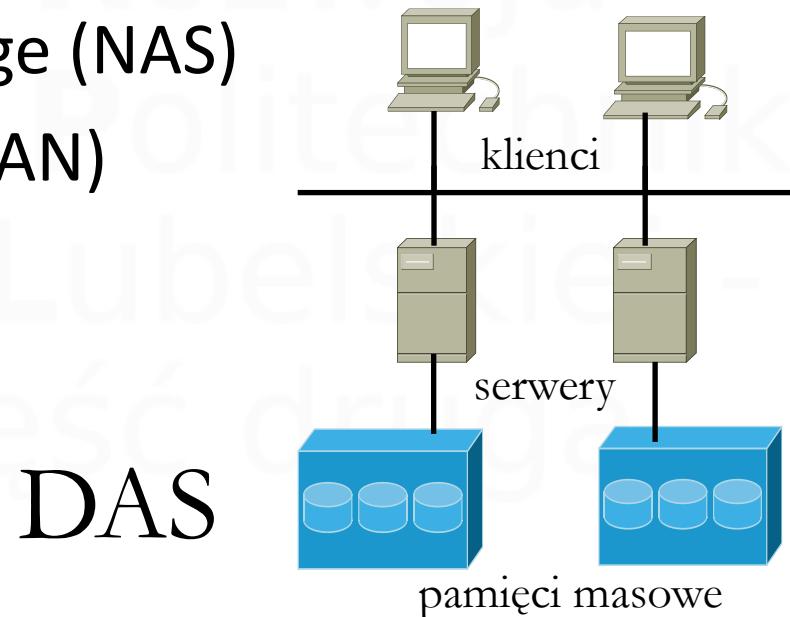


	DAS	NAS	SAN
Typ współ-dzielenia pamięci	Sektorы	Współ-dzielone pliki	Bloki
Transmisja danych	IDE/SCSI	TCP/IP, Ethernet	Fibre Channel
Dostęp	klienci lub serwery	klienci lub serwery	serwery
Pojemność (abajty)	$10^9$	$10^9 - 10^{12}$	$> 10^{12}$
Złożoność	mała	średnia	wysoka
Koszt zarządzania (na 1 GB)	wysoki	średni	niski

Źródło: Computer Desktop Encyclopedia

# Trzy typy masowych pamięci sieciowych (1)

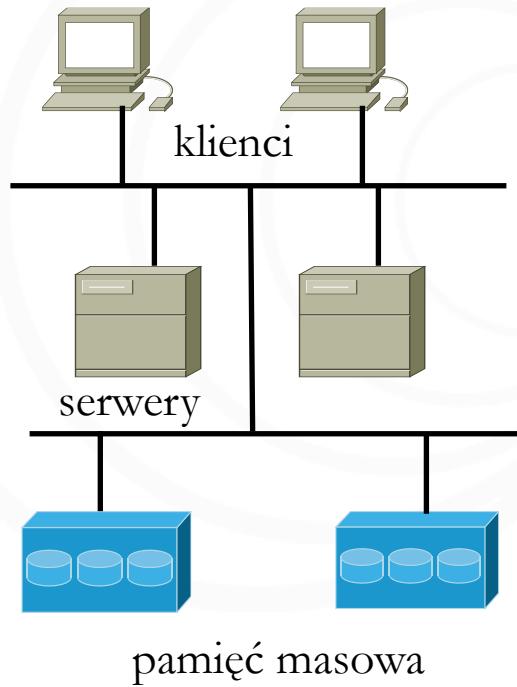
- Direct Access Storage (DAS)
  - SCSI
  - RAID
- Network Attached Storage (NAS)
- Storage Area Network (SAN)
  - Fiber Channel and
  - Fiber Channel Switch



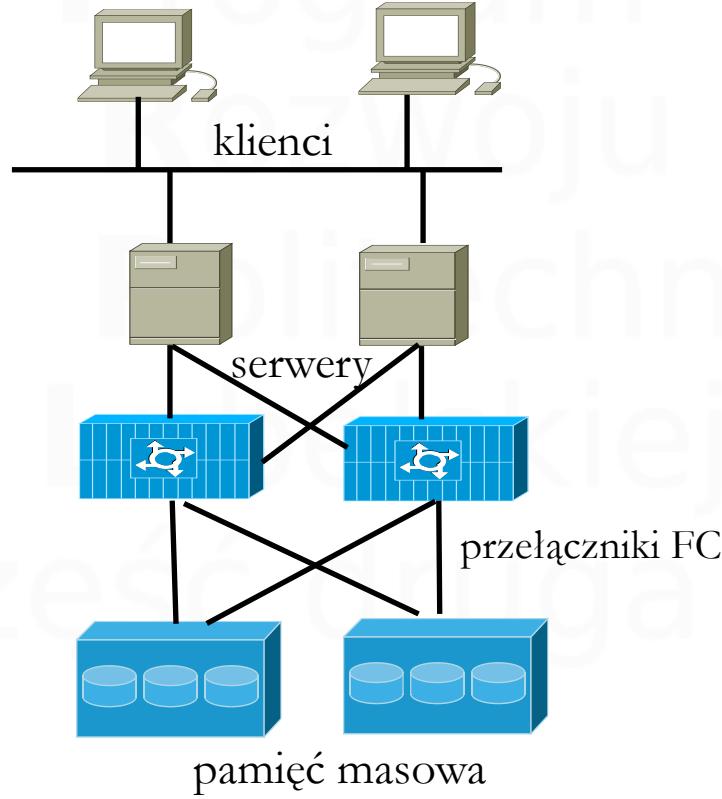
Źródło: opracowanie własne

# Trzy typy masowych pamięci sieciowych (2)

## NAS

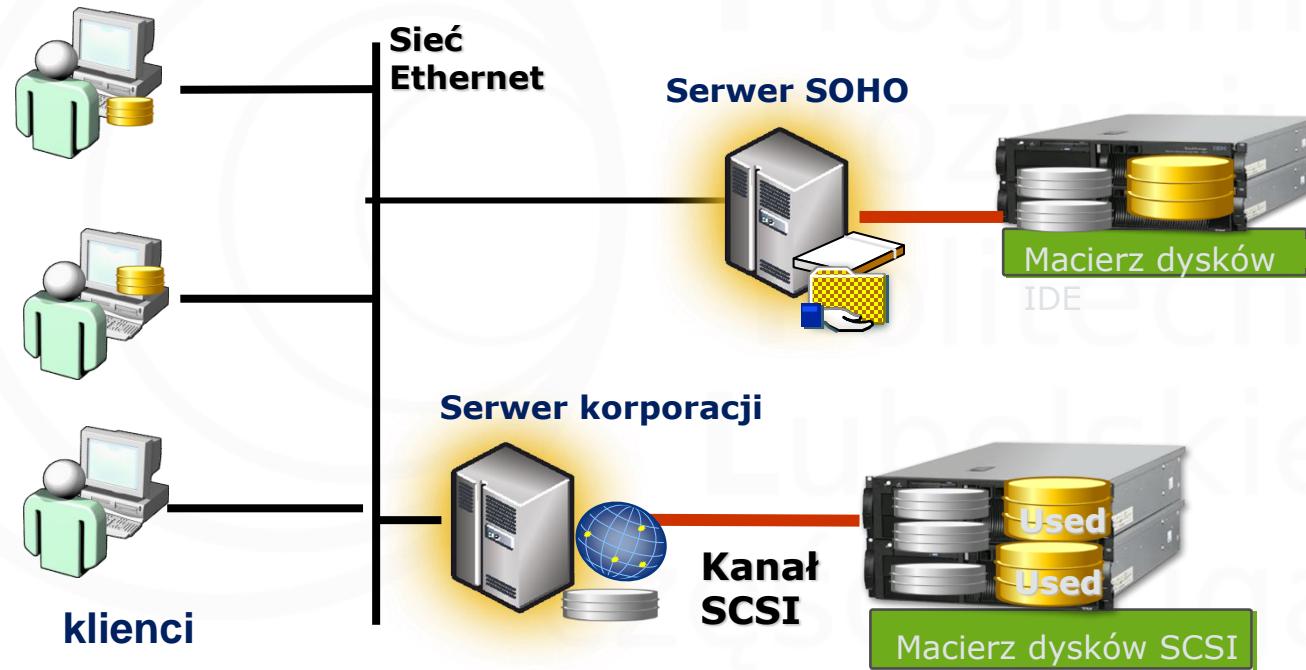


## FC-SAN



Źródło: opracowanie własne

# Pamięci DAS



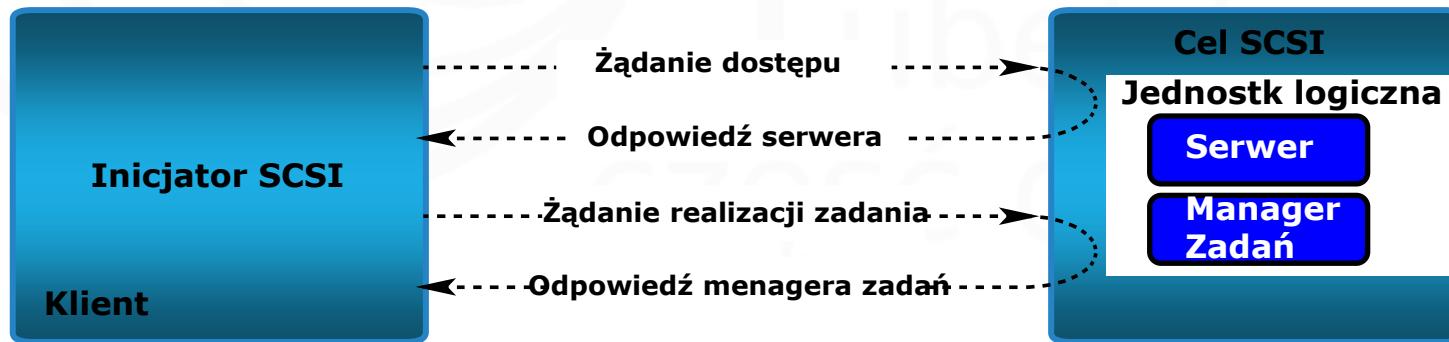
Źródło: <https://www.rfwireless-world.com/>

# Interfejs SCSI - podstawy

- Pierwotnie opracowany przez Shugart Associates i nazwany interfejsem SASI (ang. Shugart Associates System Interface)
- Wprowadzony przez ANSI jako międzynarodowy standard pod nazwą SCSI (ang. Small Computer System Interface)
- Najnowsza wersja standardu SCSI to
  - **The Ultra standard 640 SCSI**
  - **(SCSI PARALLEL INTERFACES SPI-5)**

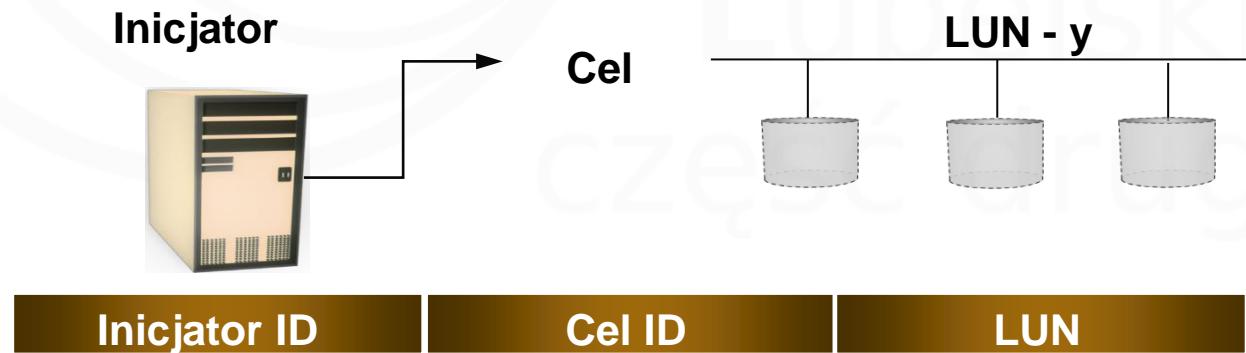
# Interfejs SCSI – architektura (2)

- SCSI wykorzystuje architekturę klient/server.
- Klient nazywany jest inicjatorem (ang. initiator) i jest implementowany jako proces I/O nadzorowany przez system operacyjny.
- Serwer jest nazywany celem (ang. target), zazwyczaj implementowany jako część oprogramowania kontrolera pamięci masowej.

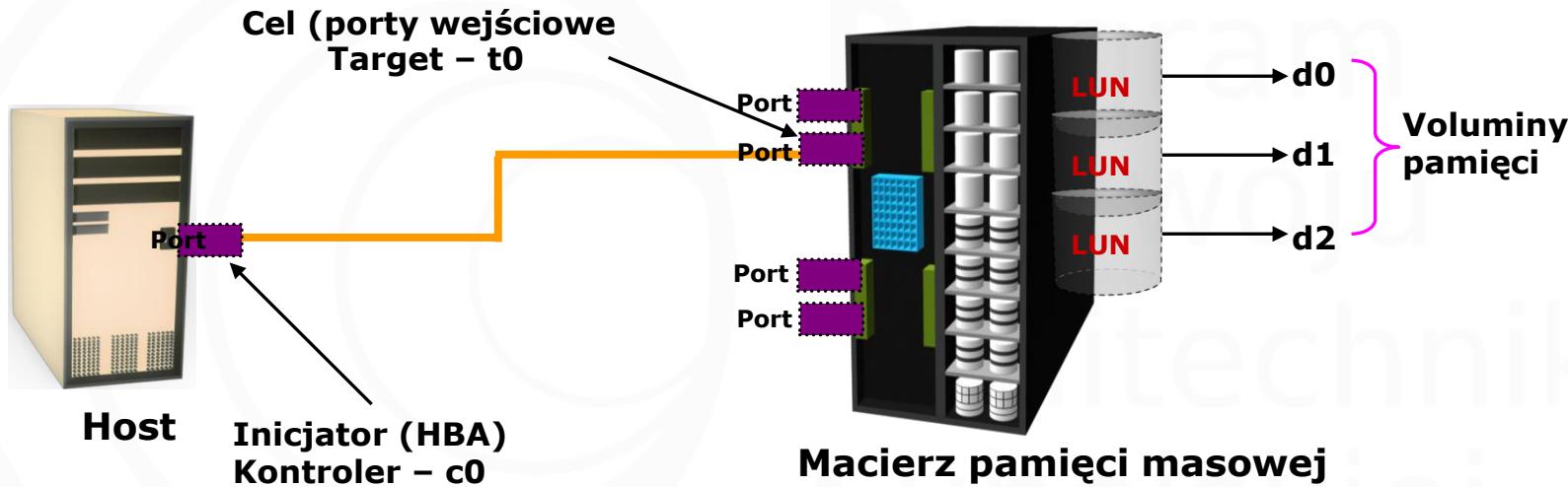


# Interfejs SCSI – adresowanie (1)

- InicjatorID – liczba od 0 do 15, typowo równa 7.
- Cel ID – liczba od 0 do 15
- LUN – liczba, która określa adresowalne urządzenie (fizyczne lub wirtualne) obsługiwane przez cel.



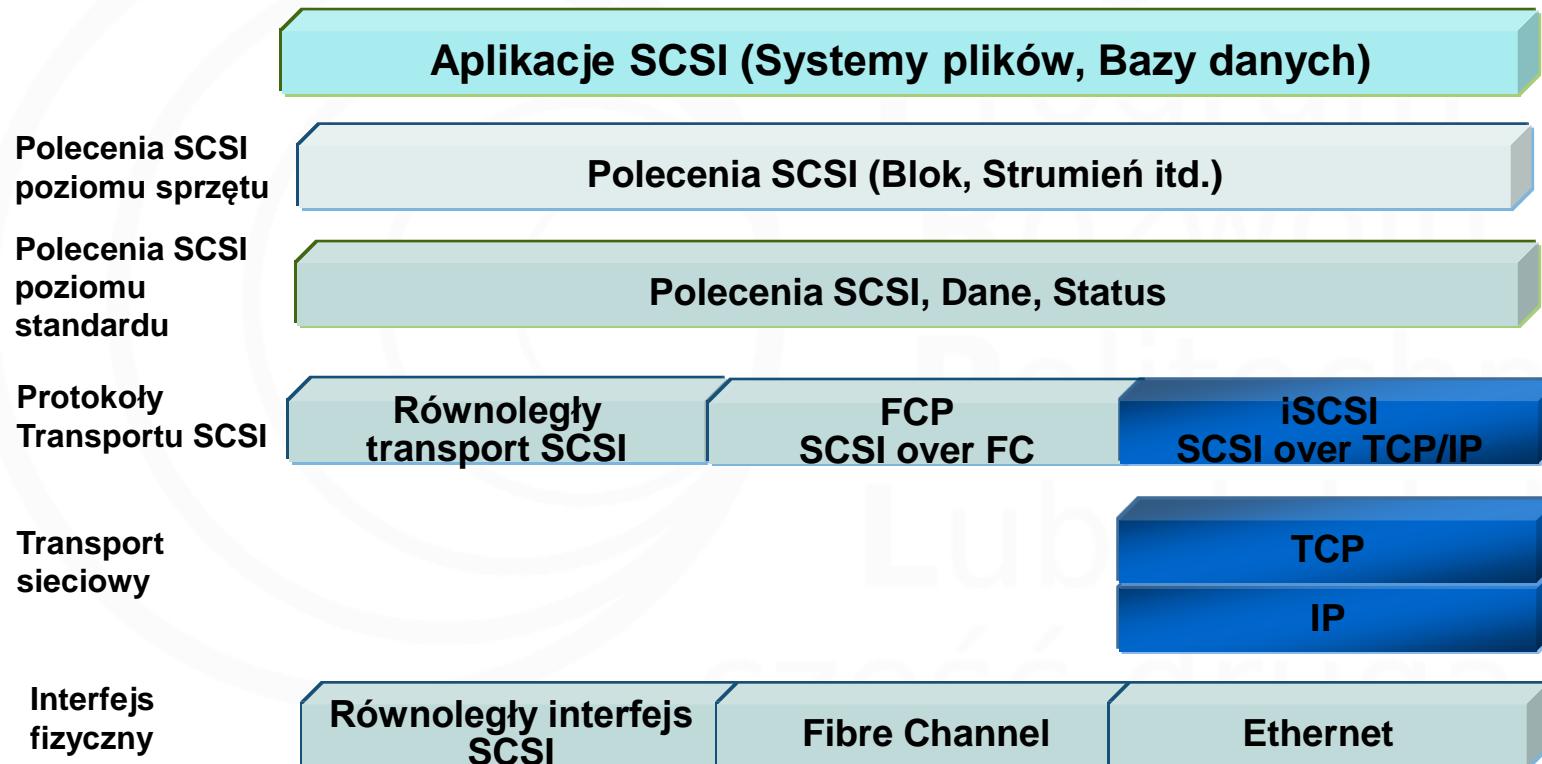
# Interfejs SCSI – adresowanie (2)



## Adresacja hosta

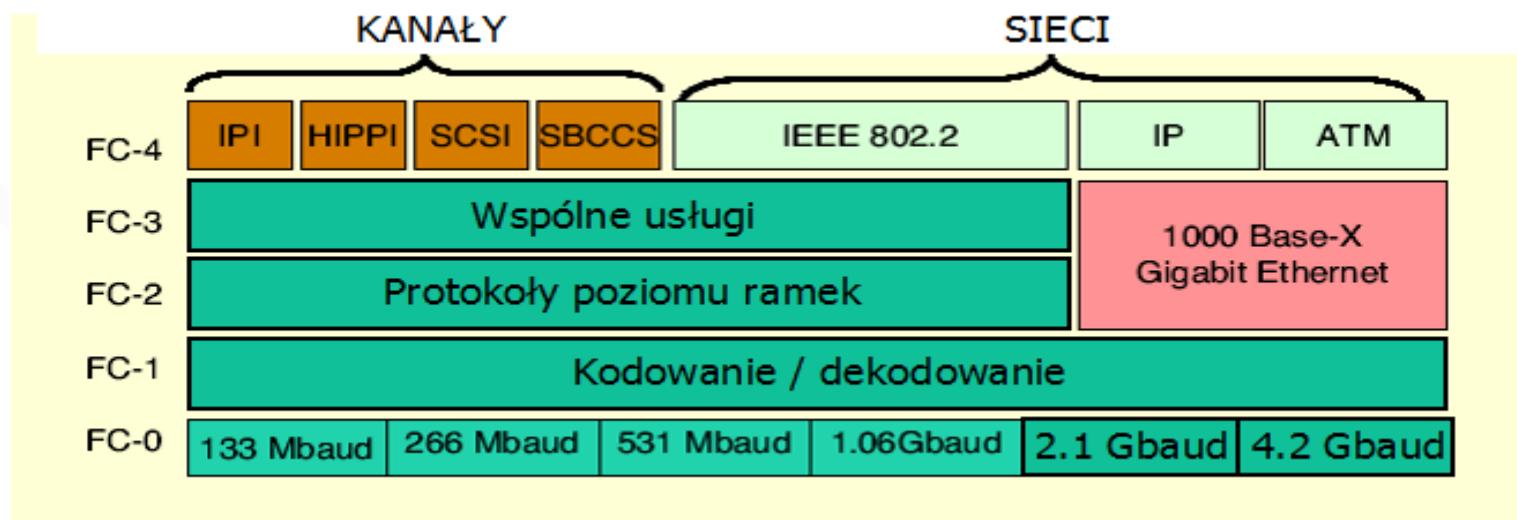
- Volumin pamięci 1 - c0t0d0
- Volumin pamięci 2 - c0t0d1
- Volumin pamięci 3 - c0t0d2

# Interfejs SCSI – rozwiązania transportowe

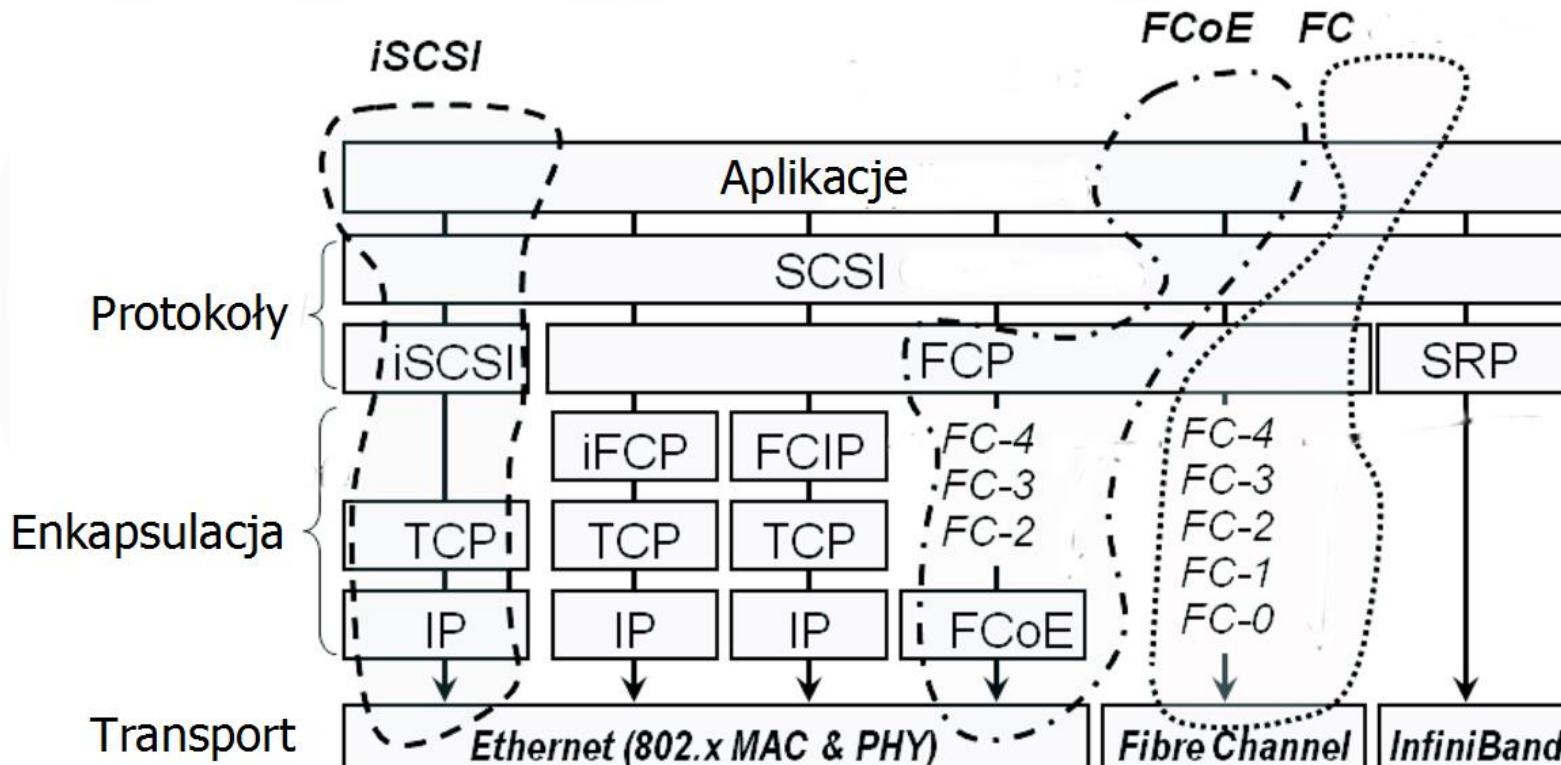


# Protokół Fibre Channel - podstawy

- FC jest protokołem kanałowo-sieciowym
  - Kanałowym: ponieważ potrafi zestawić kanały transmisji pomiędzy ograniczoną liczbą urządzeń. Ustanowiony kanał nie wymaga dalszej konfiguracji, co daje wysoką efektywność.
  - Sieciowym: ponieważ potrafi obsłużyć złożone struktury połączeń urządzeń oraz ustalać trasy pomiędzy nimi.



# Związki SCSI oraz FC z technikami sieciowymi



Źródło: opracowanie własne

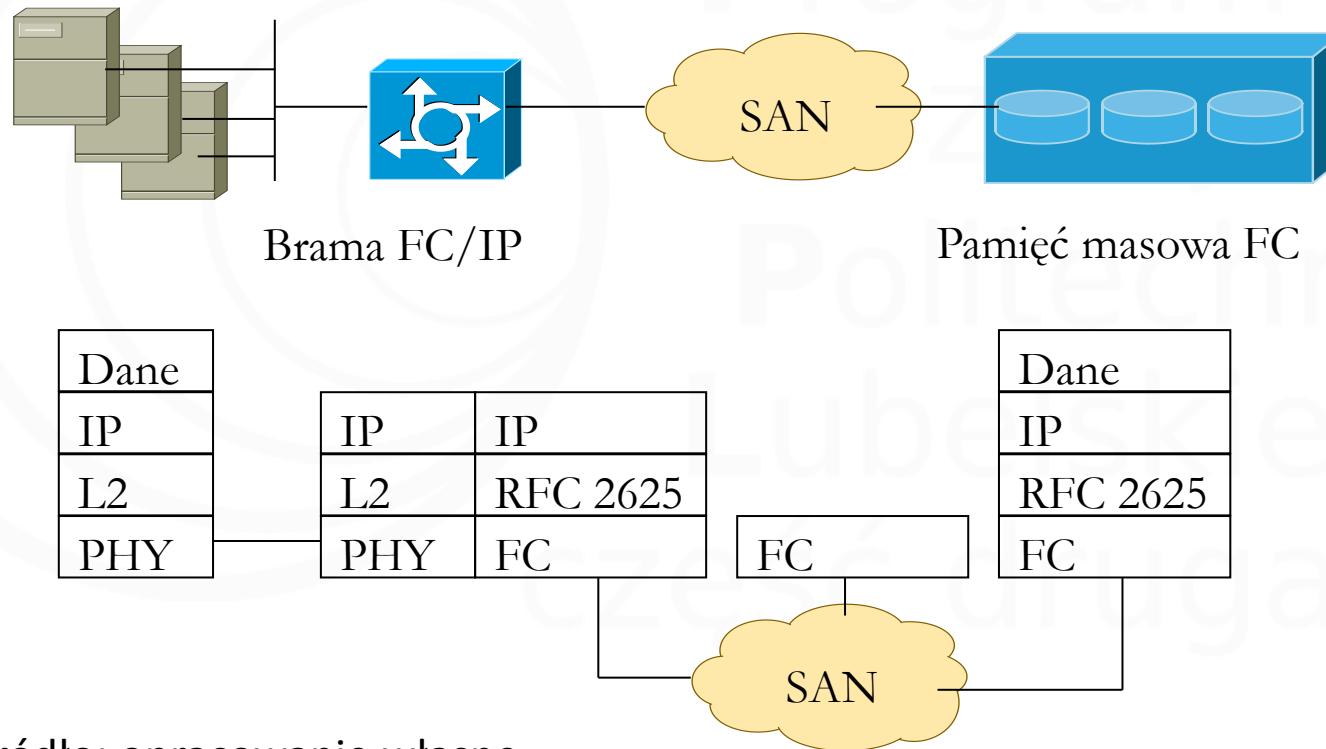
# Droga do wirtualizacji pamięci

- IP over FC (RFC 2625)
- IP-SAN
  - iSCCI (RFC 3720)
- Konwergencja IP oraz FC-SAN
  - FC Encapsulation (RFC 3643)
  - FCIP (RFC 3821) – FC over IP
  - iFCP (RFC 4172)



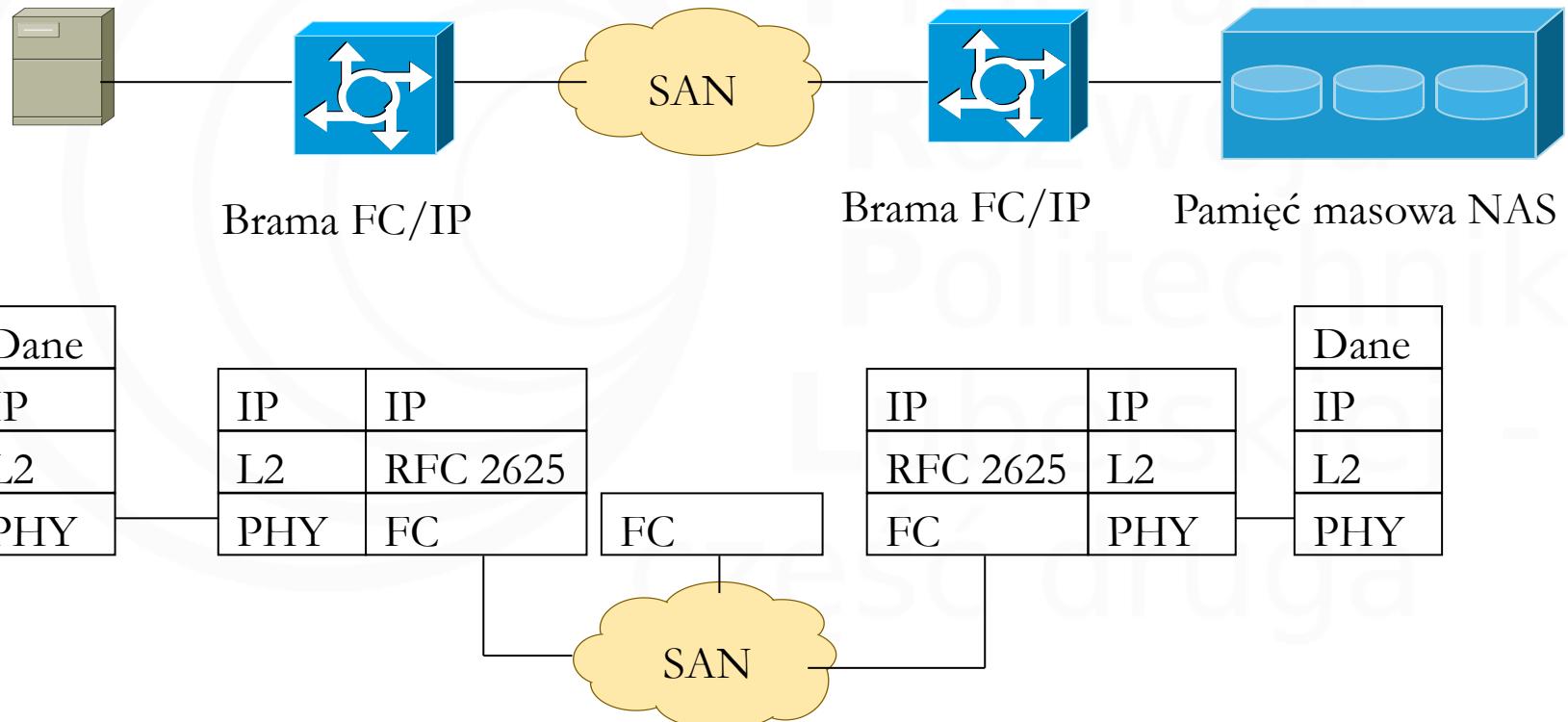
Wirtualizacja sieciowych pamięci masowych

# IPoverFC - dostęp do sieci SAN z poziomu IP



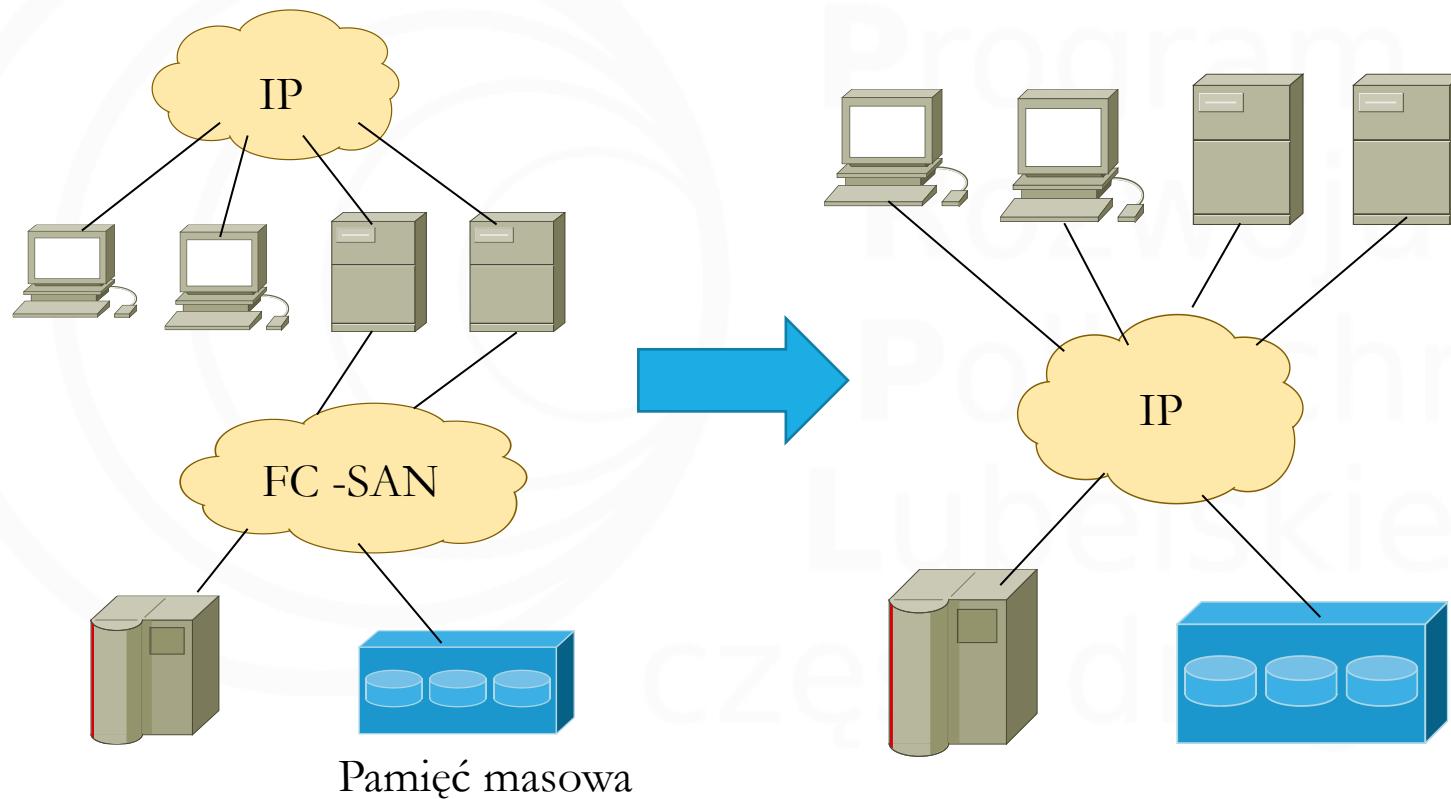
Źródło: opracowanie własne

# IPoverFC – przenoszenie danych pomiędzy NAS a SAN



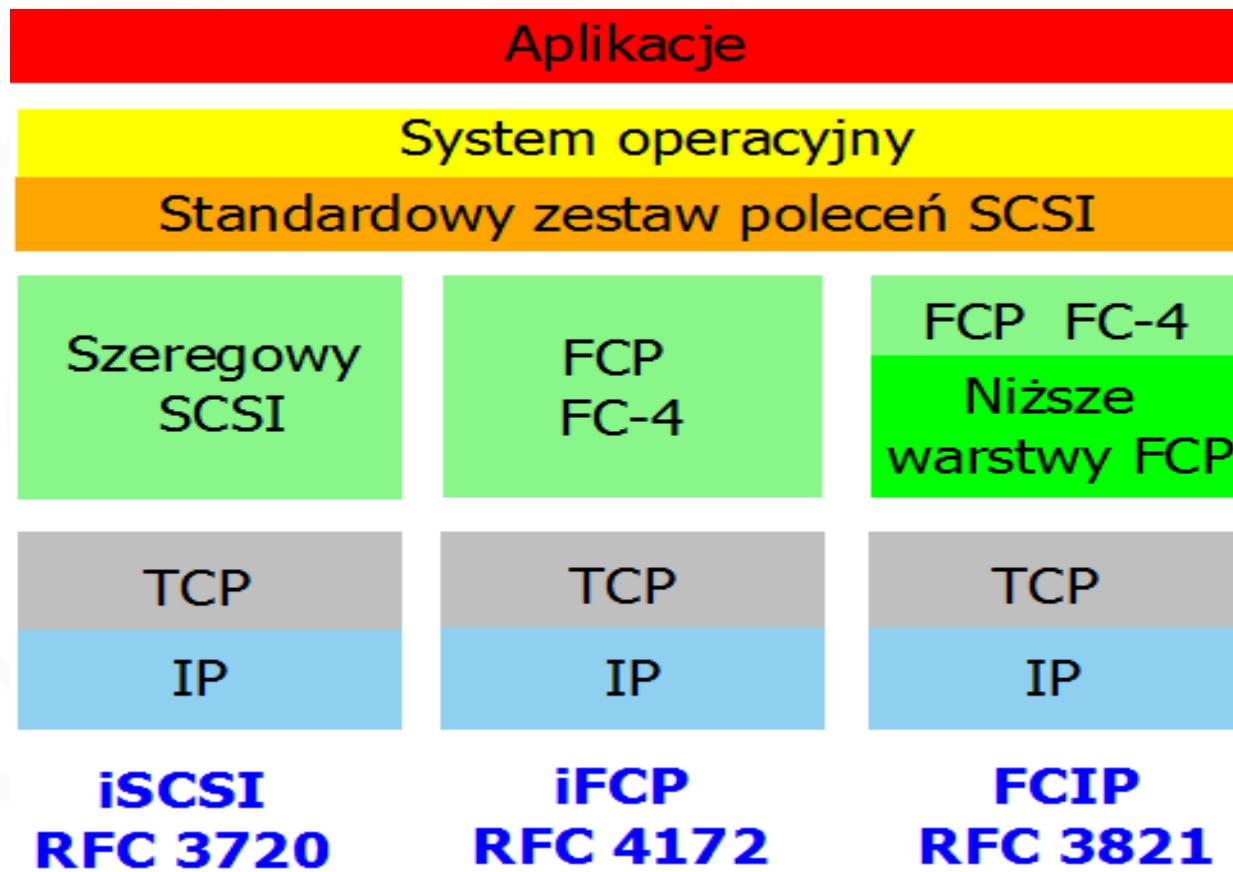
Źródło: opracowanie własne

# Kolejny krok – sieci IP-SAN



Źródło: opracowanie własne

# Protokoły IP-SAN



Źródło: opracowanie własne

# iSCSI – Adresacja i nazewnictwo (1)

- Inicjator oraz cel muszą posiadać unikalne nazwy iSCSI
  - Nazwa nie zależy od lokalizacji
  - Nazwa węzła iSCSI to nazwa urządzenia SCSI a nie adaptera
  - Zawiera do 255-bytów kodowanych UTF-8
  - Jest wykorzystywana przez SLP (ang. Service Location Protocol) V2, iSNS lub zapytania celu o nazwy
- Stosowane są dwie struktury nazw iSCSI:
  - **iqn**—kwalifikowana nazwa iSCSI
  - **eui**—Extended Unique Identifier (IEEE EUI-64—wykorzystywany również w FC WWN – Fibre Channel World Wide Name)

# iSCSI – Adresacja i nazewnictwo (2)



iqn      Type - Date - Organization Naming Authority - Subgroup Naming Authority lub String Defined by Organization Naming Authority

iqn.1987-05.com.cisco.1234abcdef987601267da232.betty  
iqn.2001-04.com.acme.storage.tape.sys1.xyz

Date = yyyy-mm  
Data otrzymania domeny

Zarezerwowana nazwa domenowa

eui      Type - EUI-64 Identifier (ASCII Encoded Hexadecimal)

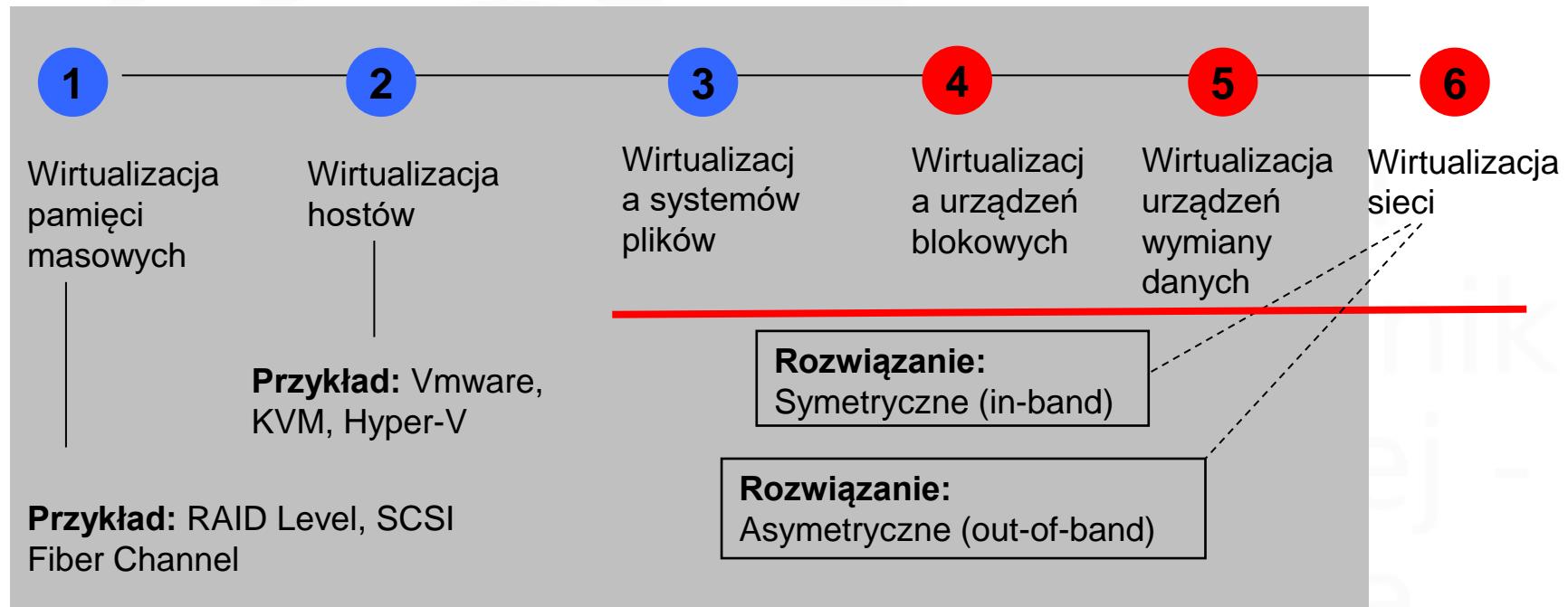
eui.02004567a425678d

Źródło: opracowanie własne

# Wirtualizacja pamięci masowych

- Celem wirtualizacji pamięci masowych jest dostarczenie użytkownikom dodatkowych poziomów abstrakcji, które zredukują złożoność procesu użytkowania i zarządzania pamięcią.
- Próby realizacji powyższego celu doprowadziły do konieczności koordynacji działań wielu ośrodków badawczych i firm z branży IT.
- Z tego powodu powołano do życia  
**SNIA (ang. Storage Networking Industry Association)**

# Wirtualizacja według SNIA



Źródło: opracowanie własne



## Programowanie aplikacji w chmurze obliczeniowej

## Rozwiązania chmurowe w systemach Linux

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni



**Fundusze  
Europejskie**  
Wiedza Edukacja Rozwój



**Rzeczpospolita  
Polska**

**Unia Europejska**  
Europejski Fundusz Społeczny



# Historia – wirtualizacja

- **Chroot** (wersja 7 Unix, 1979)
- **FreeBSD Jails** (FreeBSD 4, 2000)
- **Linux vserver** (Linux, Październik 2001)
- Parawirtualizacja Xen (Linux, 2003)
- **Solaris zones** (Solaris 10, 2004)
- **OpenVZ** (Linux, 2005)
- Pełna wirtualizacja KVM (Linux, 2007)
- **Linux Containers** - LXC (Linux 2.6.29 2009)

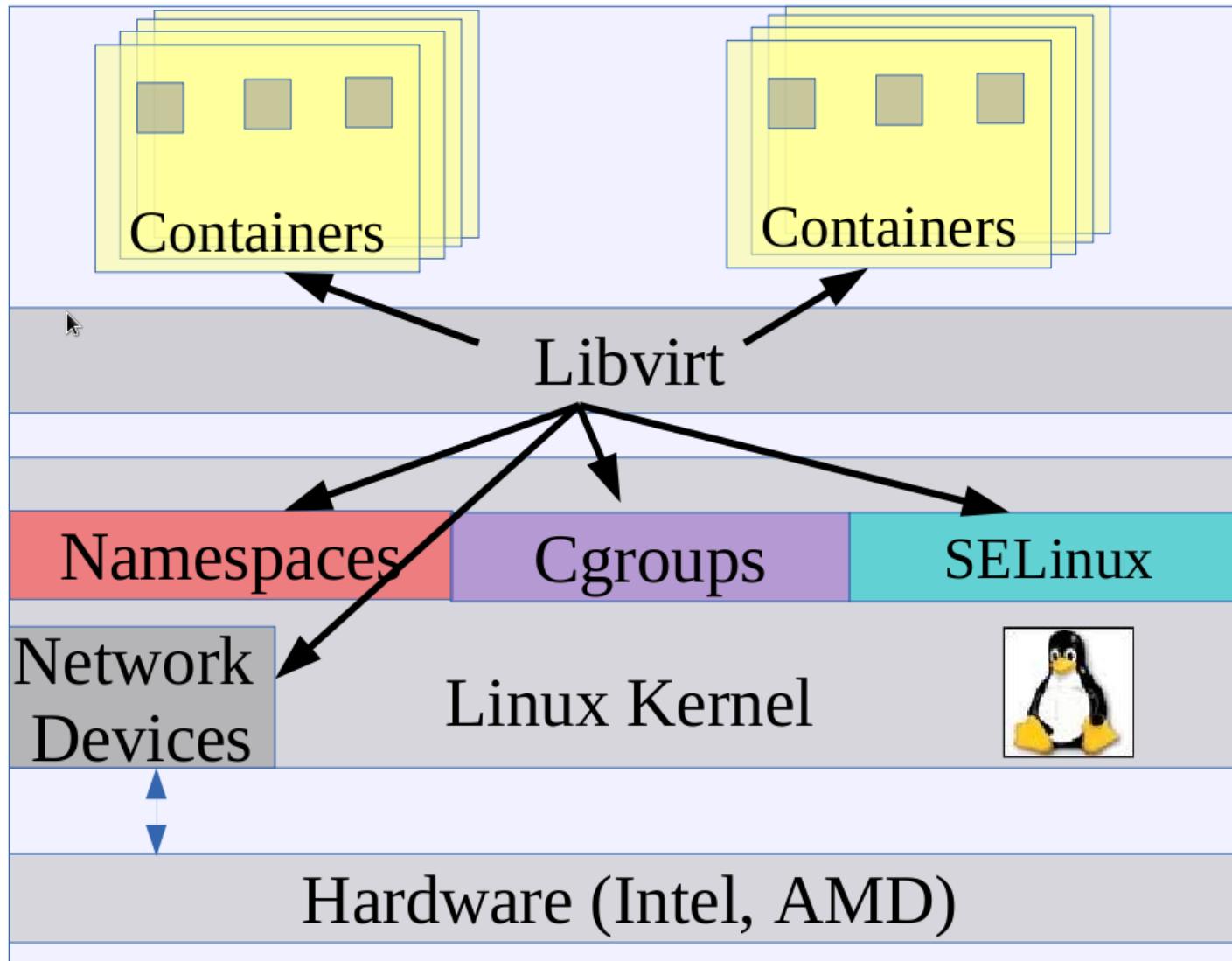
**W kolorze** – Wirtualizacja na poziomie systemu operacyjnego (kontenery)

# LXC – co to takiego?

- Wirtualizacja na poziomie systemu operacyjnego
  - lekka wirtualizacja
  - kontenery
  - bazuje na cgroup, selinux i przestrzeni nazw (namespace)
  - włączona w jądro
- Może być zarządzana przez libvirt-lxc lub lxc-tools
- Wydajność na poziomie bare metal

# LXC – co to takiego?

- Lepsza izolacja niż chroot (chroot jail)
- Mały narzut, LXC używają minimalnych zasobów RAM/HDD w porównaniu do maszyn wirtualnych
- Aplikacje i serwisy działają z natywną prędkością
- Istnieje wsparcie dla kontenerów Linux w libvirt
- Współpracują dobrze z btrfs .
- Działają na 32 i 64 bitowych procesorach
- Kontenery Linux-owe są Open source.
- Nie wymagają łaty na jądro w przeciwieństwie do XEN czy OpenVZ



Źródło: Red Hat – lekka wirtualizacja

# Możliwe użycie kontenerów

- Niewymagające serwery web
- Środowiska do testowania
- Izolacja aplikacji
- Aplikacje wymagające małych opóźnień
- Możliwość użycia w dokerach

# Słabości

- Zamknięte w uruchomionym jądrze hosta
  - W przeciwieństwie do w pełni zwirtualizowanych maszyny, jest ograniczony do działającego jądra na hoście
- Brak wsparcia dla MS Windows
  - jakkolwiek jest możliwe uruchomienie wine w kontenerze/dokerze

# Kontenery LXC w systemie Ubuntu z klientem CentOS

- Należy utworzyć grupę kontrolną systemu plików jako root

```
# mkdir -p /cgroup  
# mount none -t cgroup /cgroup
```

Dodać ją do fstab tak aby zmiany były aktywne po restarcie

```
# edit /etc/fstab i dodać poniższą linijkę  
none /cgroup cgroup defaults 0 0
```

- Pakiety do zainstalowania

```
# apt-get install libvirt-bin debootstrap
```

- Należy pobrać i zainstalować paczkę

lxc\_0.7.2-1\_amd64.deb

```
# dpkg -i lxc_0.7.2-1_amd64.deb
```

# Krok 1: utworzenie kontenera

- # Utwórz kontener  
`# lxc-create -f  
/sciezka/do/{NAZWA_KONTENRA}/main/config/file -n  
{NAZWA_KONTENERA}`  
`# lxc-create -f /etc/lxc/lxc-centos.conf -n  
centos`  
#Utwórz katalog rootfs  
`# mkdir /var/lib/lxc/centos/rootfs`
- Skopiuj wcześniej przygotowane pliki  
`# cp -r /home/wojekatalogdomowy/LXC/Centos5-  
x86_64/* /var/lib/lxc/{NAZWA_KONTENERA}/rootfs`  
`# cp -r /home/wojekatalogdomowy/LXC/Centos5-  
x86_64/* /var/lib/lxc/centos/rootfs`
- Utwórz skrypt konfiguracyjny dla domyślnych nastaw kontenera  
`# edit /usr/local/bin/lxc-config`
- Dla nowego kontenera skrypt ma być uruchamiany tylko raz

# Krok 1a: lxc-config

```
#!/bin/bash
# bodhi.zazen's lxc-config
# Makes default devices needed in lxc
# containers
# modified from http://lxc.teegra.net/
ROOT=$(pwd)
DEV=${ROOT}/dev
if [ $ROOT = '/' ]; then
printf "\033[22;35m\nDO NOT RUN ON THE
HOST NODE\n\n"
tput sgr0
exit 1
fi
if [ ! -d $DEV ]; then
printf "\033[01;33m\nRun this script in
rootfs\n\n"
tput sgr0
exit 1
fi
rm -rf ${DEV}
mkdir ${DEV}
mknod -m 666 ${DEV}/null c 1 3
mknod -m 666 ${DEV}/zero c 1 5
mknod -m 666 ${DEV}/random c 1 8
mknod -m 666 ${DEV}/urandom c 1 9
mkdir -m 755 ${DEV}/pts
mkdir -m 1777 ${DEV}/shm
mknod -m 666 ${DEV}/tty c 5 0
mknod -m 666 ${DEV}/tty0 c 4 0
mknod -m 666 ${DEV}/tty1 c 4 1
mknod -m 666 ${DEV}/tty2 c 4 2
mknod -m 666 ${DEV}/tty3 c 4 3
mknod -m 666 ${DEV}/tty4 c 4 4
mknod -m 600 ${DEV}/console c 5 1
mknod -m 666 ${DEV}/full c 1 7
mknod -m 600 ${DEV}/initctl p
mknod -m 666 ${DEV}/ptmx c 5 2
exit 0
```

# Krok 2: Sesja chroot

- ```
# cd /var/lib/lxc/{NAZWA KONTENERA}/rootfs/
# cd /var/lib/lxc/centos7/rootfs/
# /usr/local/bin/lxc-config
# fix /dev
```
- Zmiana systemu na Centos  

```
# chroot /var/lib/lxc/centos/rootfs/
Ustawienie hasła w kontenerze
# passwd root
```
- Dodanie/usunięcie usług  

```
# chkconfig sshd on
# chkconfig httpd on
# chkconfig xinetd off
# chkconfig sendmail off
```
- Aby skończyć sesję polecenie **exit**

# Krok 3: Usuwanie kontenera

- Usuwanie kontenera i plików  
`# lxc-destroy -n {NAZWA_KONTENERA}`  
`# lxc-destroy -n centos`
- Start kontenera  
`# lxc-start -n {NAZWA_KONTENERA} init`  
`# lxc-start -n centos init`
- Zatrzymanie kontenera  
`# lxc-stop -n {NAZWA_KONTENERA}`  
`# lxc-stop -n centos`

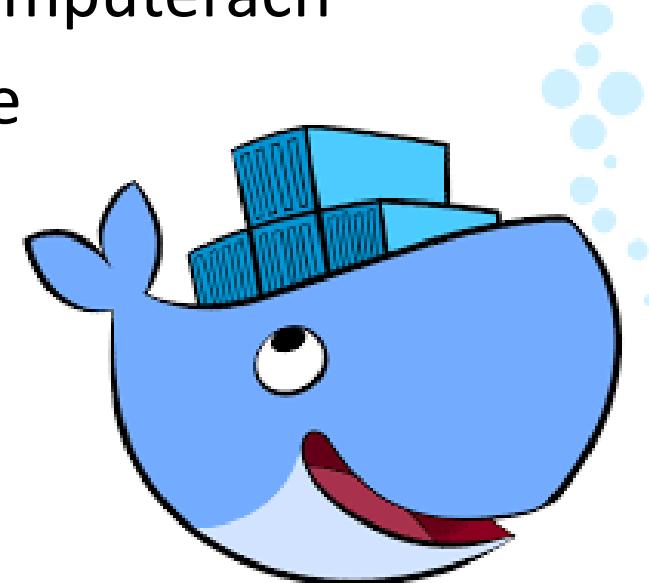
- Połączenie do konsoli tty  
`# lxc-console -n centos -t`
- Sprawdzenie konfiguracji  
`# lxc-checkconfig`
- Wyświetlenie kontenerów  
`# lxc-ls`  
W przypadku komunikatu o błędzie lxc-ls „got bogus unix line”, należy :  
`# sudo apt-get --reinstall install uml-utilities`
- Bieżący status  
`# lxc-info -n {CONTAINER_NAME}`  
`# lxc-info -n centos`

# Zalety i wady kontenerów

| Zalety                  | Wady                        |
|-------------------------|-----------------------------|
| 1. Zredukowany narzut   | 1. Ograniczona elastyczność |
| 2. Większe zagęszczenie | 2. Zmniejszona izolacja     |
| 3. Zredukowany rozrost  |                             |

# DOCKER

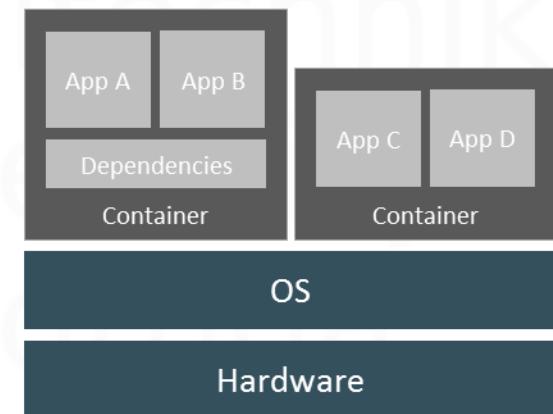
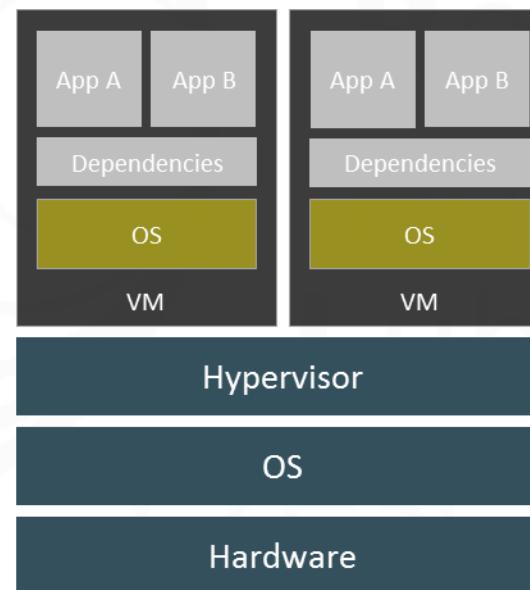
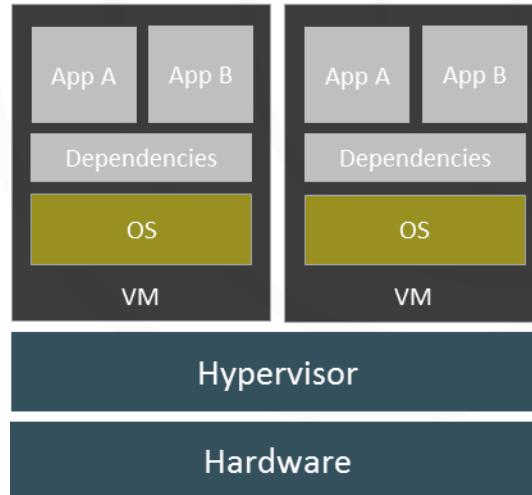
- Narzędzie bazujące na kontenerach
- Instrument wysokiego poziomu dla LXC
- Przenośny sposób wdrażania na komputerach
- Kontenery publicznie udostępniane
- Automatyczna budowa
- Ekosystem narzędzi (nova, salt, chef, puppet, ansible jenkins, openshift ...)
- Więcej na: <http://docker.io>



Źródło: docker.io

# Technologie wirtualizacji

- Kontenery są lekkie:
  - współdzielą jądro systemu na hoście
  - współdzielą główny system plików systemu gospodarza w stosownych przypadkach



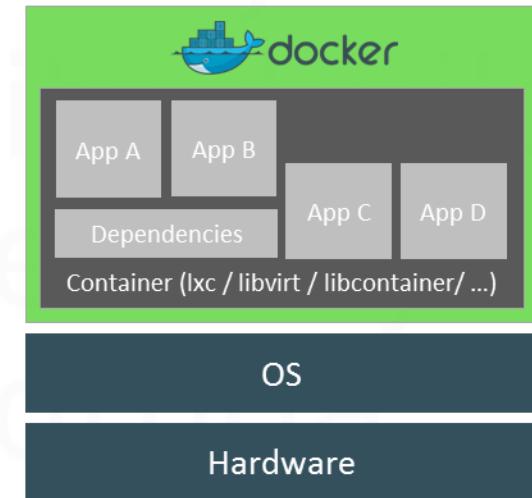
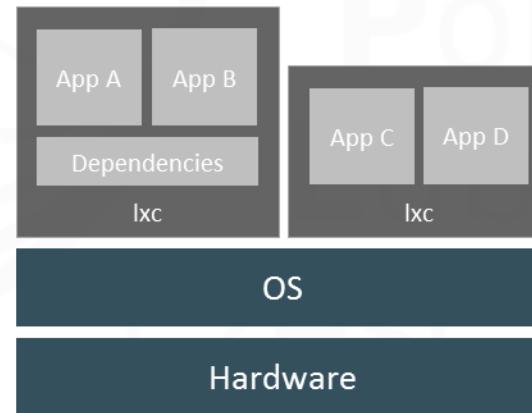
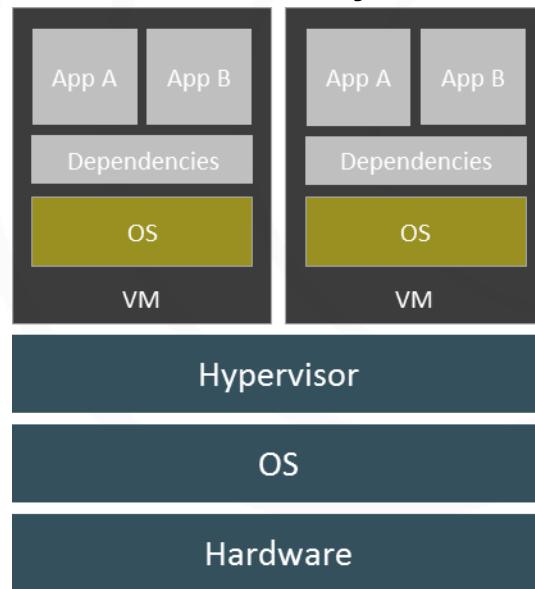
**Type 1 Hypervisor**

Źródło: Paul Farmer, Virtualization Redefined, MontaVista Software

**Type 2 Hypervisor**

# Technologie wirtualizacji

- Docker zapewnia zunifikowany dostęp do:
  - Technologii kontenerów w systemie Linux (cgroups, namespaces)
  - Różnych implementacji kontenerów (lxc, libvirt, libcontainer, itp.)
- ‘libcontainer’ jest dokerową implementacją technologii kontenerów



Type 2 Hypervisor

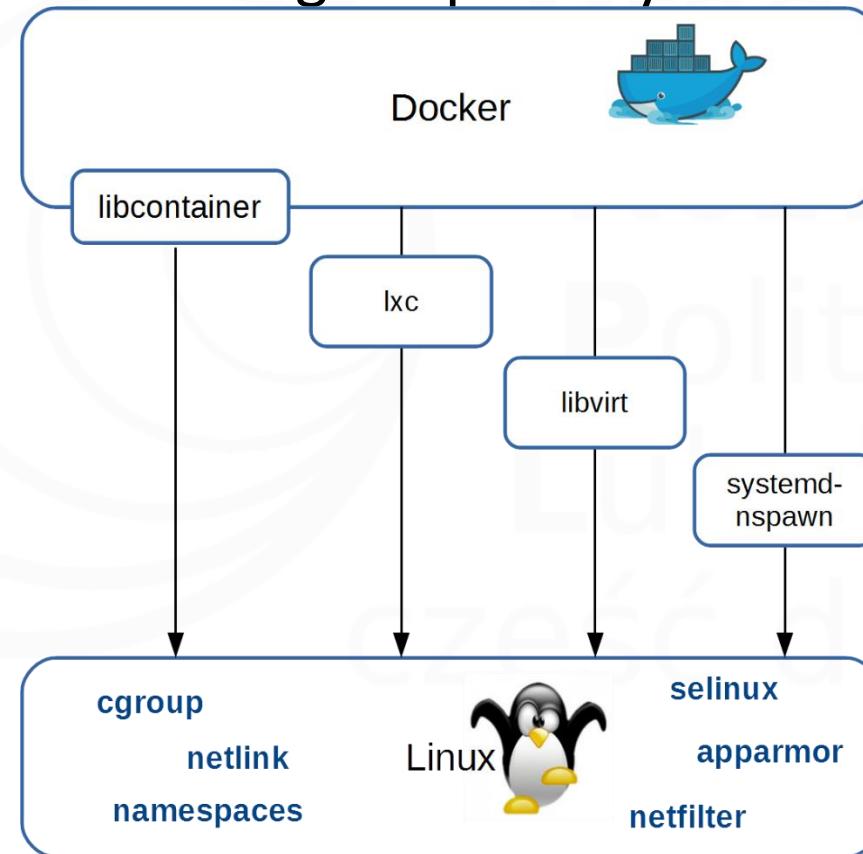
LXC

Docker

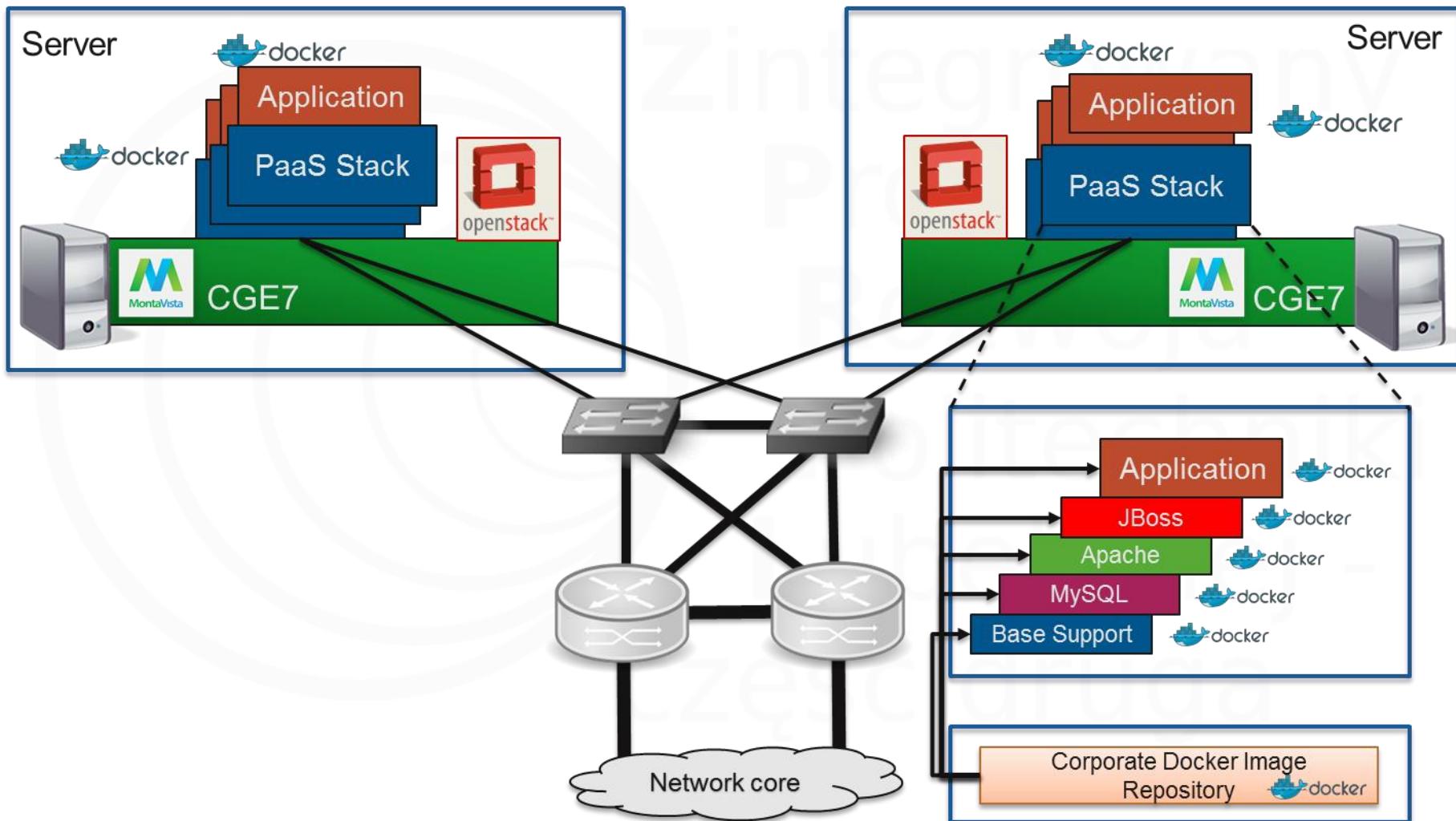
Źródło: Paul Farmer, Virtualization Redefined, MontaVista Software

# Doker a kontenery

- Docker – technologia u podłoży

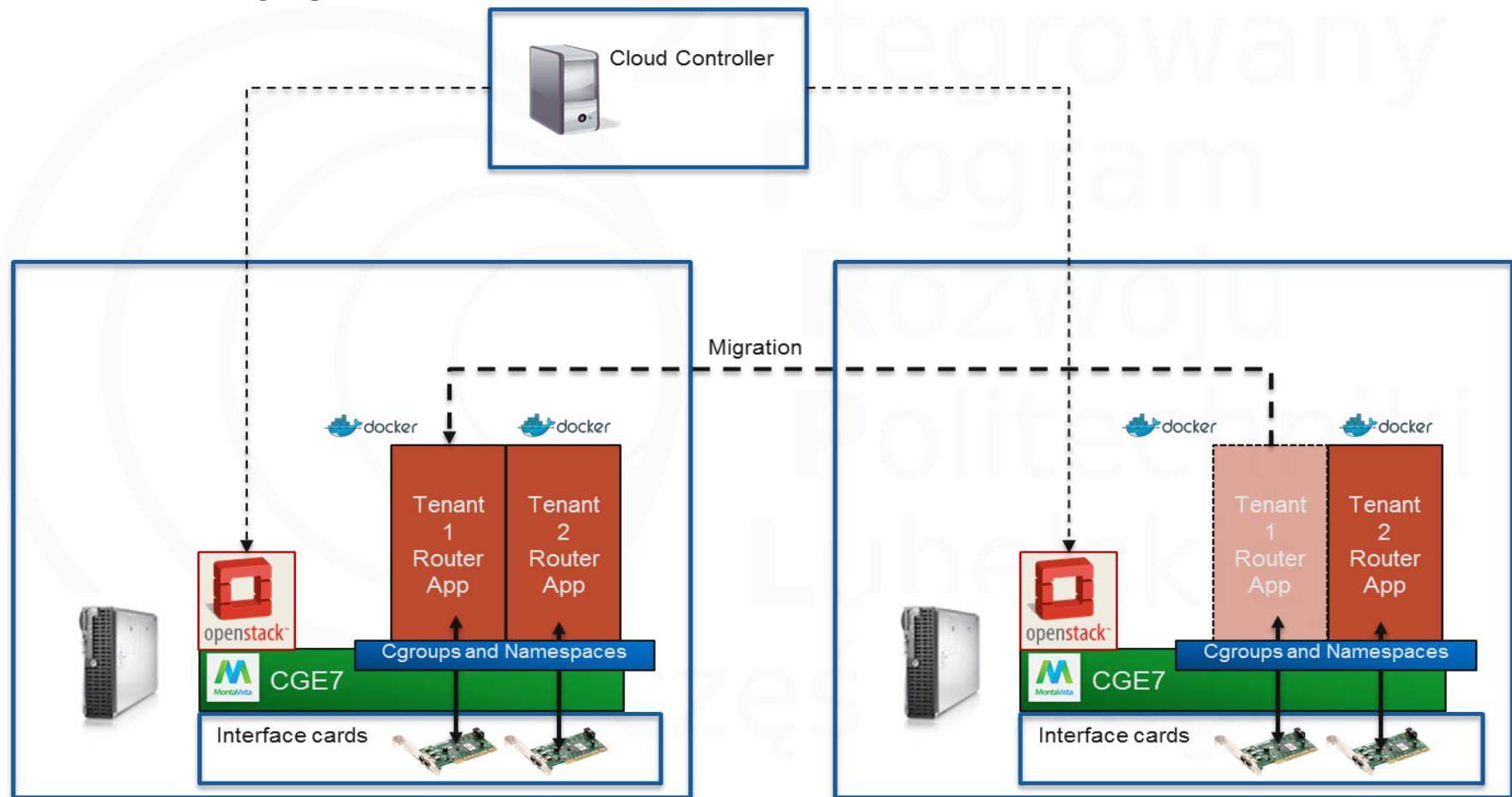


# Docker – chmura PaaS



Źródło: Paul Farmer, Virtualization Redefined, MontaVista Software

# Wielo-dzierżawcość w chmurze bazująca na kontenerach



Źródło: Paul Farmer, Virtualization Redefined, MontaVista Software



## Programowanie aplikacji w chmurze obliczeniowej

**Paradygmat programowania równoległego  
MapReduce, system Hadoop jako przykład  
programowania w chmurze**

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni



**Fundusze  
Europejskie**  
Wiedza Edukacja Rozwój



**Rzeczpospolita  
Polska**

**Unia Europejska**  
Europejski Fundusz Społeczny



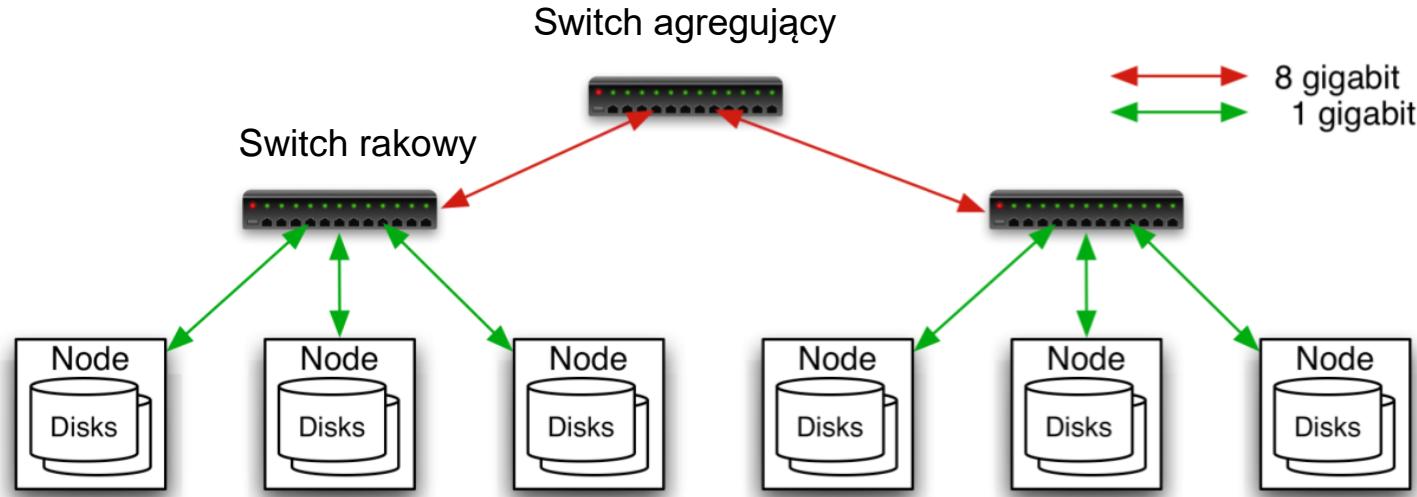
# Hadoop - wprowadzenie

- Potrzeba do przetwarzania ogromnych zbiorów danych na dużych klastrach komputerów
- Bardzo droga budowa niezawodności w każdej aplikacji
- Codzienne awarie węzłów
  - Awarie są spodziewane, aniżeli wyjątkowe
  - Liczba węzłów w klastrze nie jest stała
- Potrzebna wspólna infrastruktura
  - Wydajne, niezawodne, łatwe w użyciu
  - Otwarto-źródłowe, Licencja Apache

# Rozwiązania Hadoop

- Amazon/A9
- Facebook
- Google
- New York Times
- Veoh
- Yahoo!
- .... wiele innych

# Powszechny sprzęt



- Typowo w 2-warstwowej architekturze
  - Węzły to ogólniedostępne PCs
  - 30-40 węzłów/szafę
  - Uplink z szafy to 3-4 Gigabyty/s
  - Wewnętrzna przeływnośc w szafie 1 Gb/s

# Zalety HDFS

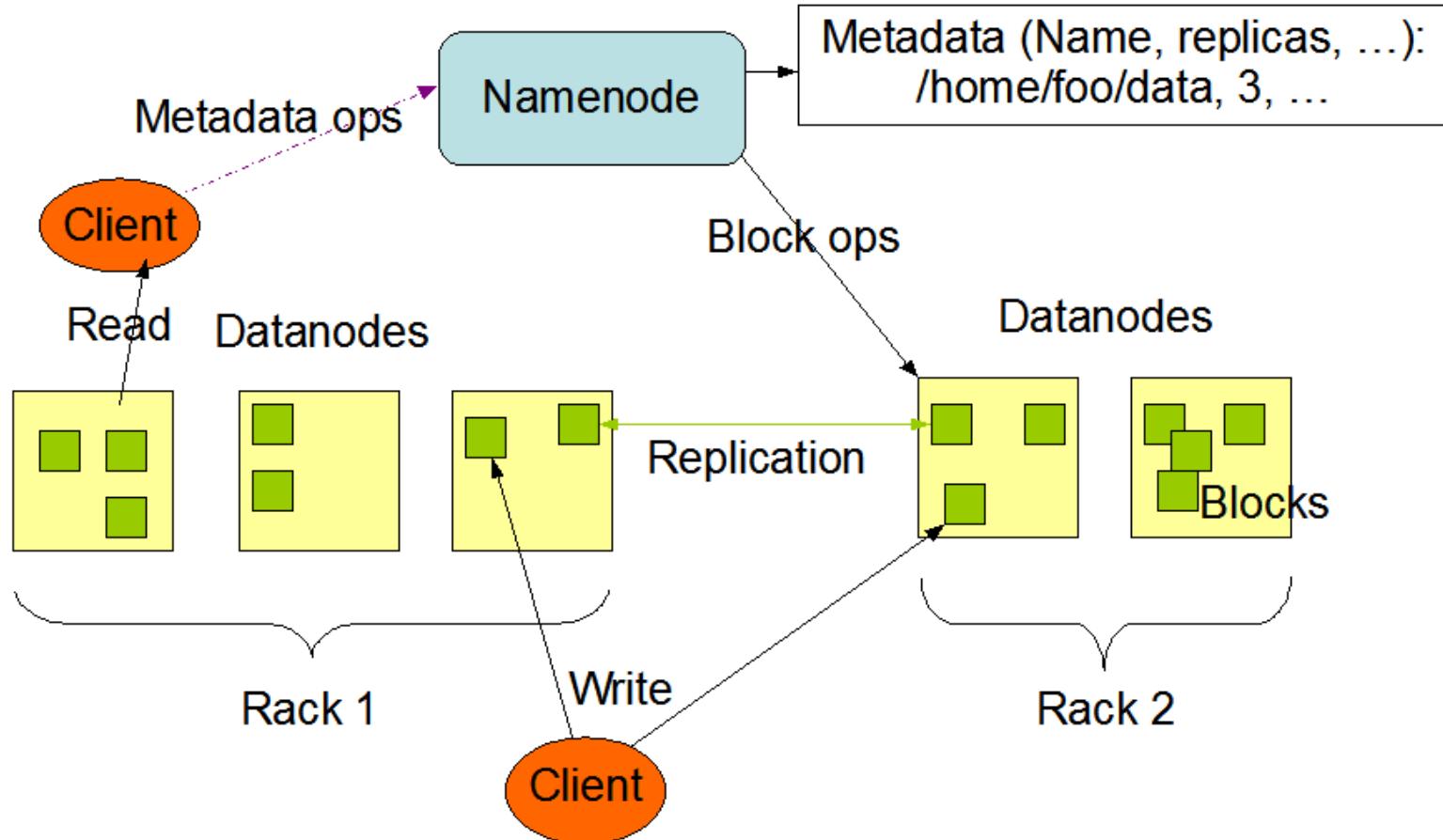
- Bardzo duży rozproszony system plików
  - 10K węzłów, 100 millionów plików, 10PB
- Działa na ogólnodostępnym sprzęcie
  - Pliki są replikowane aby obsłużyć przypadki awarii
  - Wykrywa awarie i jest w stanie sobie z nimi radzić
- Zoptymalizowany do przetwarzania wsadowego
  - Lokalizacje danych odsłonięte tak, że obliczenia mogą się odbywać w miejscu przechowywania danych
  - Zapewnia bardzo wysoką zagregowaną przepustowość



# Rozproszony system plików

- Pojedyncza przestrzeń nazw dla całego klastra
- Spójność danych
  - Model dostępu write-once-read-many
  - Klient może tylko dopisywać do istniejącego pliku
- Pliki są podzielone na bloki
  - Typowy rozmiar bloku to 64MB
  - Każdy blok jest powielany na wielu węzłach danych
- Inteligentny klient
  - Klient może odnaleźć lokalizację bloku
  - Klient uzyskuje dostęp do danych z węzła danych

# Architektura HDFS



Źródło: <https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html>

# Węzeł główny (NameNode)

- Zarządzalna przestrzeń nazw systemu plików
  - Mapuje nazwę pliku do zestawu bloków
  - Mapuje bloki do węzłów danych (DataNode) w miejscu ich fizycznego składowania
- Zarządzanie konfiguracją klastra
- Silnik replikacji dla bloków

# Węzeł główny Metadane

- Metadane w pamięci
  - Bieżące metadane są w pamięci głównej
  - Brak zapotrzebowania stronicowania metadanych
- Typy metadanych
  - Lista plików
  - Lista bloków dla każdego pliku
  - Lista węzłów z danymi dla każdego bloku
  - Atrybuty plików, np. czas utworzenia, wsp. replikacji
- Log transakcyjny
  - Zapisuje tworzenie i usuwanie plików itp.

# Węzeł zarządzający danymi (DataNode)

- Serwer alokacji bloków
  - Przechowuje dane w lokalnym systemie plików (np. ext3)
  - Przechowuje metadane bloku (np. CRC)
  - Zwraca dane i metadane do klientów
- Raport bloków
  - Okresowo wysyła raport wszystkich istniejących bloków do węzła głównego (NameNode)
- Ułatwia przetwarzanie potokowe danych
  - Przekazuje dane do innych określonych węzłów danych

# Umiejscowienie bloków

- Bieżąca strategia
  - Jedna replika na węźle lokalnym
  - Druga replika w szafie zdalnej
  - Trzecia replika w tej samej zdalnej szafie
  - Dodatkowe repliki są umiejscawiane losowo
- Klienci odczytują z najbliższej repliki

# Narzędzia HDFS

- Tętno (Heartbeats)
  - Węzły z danymi (DataNodes) wysyłają potwierdzenie do węzła głównego (NameNode)
    - raz na 3 sekundy
  - Węzeł główny (NameNode) używa potwierdzeń do wykrycia awarii węzłów z danymi (DataNode)
- Silnik replikacji
  - Węzeł główny (NameNode) wykrywa awarie węzłów z danymi
    - Wybiera nowe węzły z danymi do ponownej replikacji
    - Balansuje użycie dysku
    - Równoważy ruch komunikacyjny do węzłów danych

# Awaria węzła głównego (NameNode)

- Pojedynczy punkt awarii
- Logi z transakcjami składowane w różnych katalogach
  - Katalog w lokalnym systemie plików
  - Katalog w zdalnym systemie plików (NFS/CIFS)

# Przetwarzanie potokowe danych

- Klient otrzymuje listę węzłów danych, na których można umieścić repliki bloku
- Klient zapisuje do pierwszego węzła DataNode
- Pierwszy DataNode przesyła dane do następnego węzła w potoku
- Gdy wszystkie repliki są napisane, klient przechodzi do następnego bloku zapisu w pliku

# Wyrównywacz (Rebalancer)

- Cel: procent zapełnienia dysków DataNodes powinien być podobny
  - Zwykle uruchamiany, gdy nowe węzły danych są dodawane.
  - Klaster jest w trybie online, kiedy Rebalancer jest aktywny
  - Rebalancer jest dławiony, aby uniknąć przeciążenia sieci
  - Narzędzia wiersza poleceń

# Zapasowy węzeł główny (NameNode)

- Kopiuje FslImage i log transakcji z węzła głównego do katalogu tymczasowego
- Łączy FSImage i log w nową strukturę FSImage w katalogu tymczasowym
- Aktualizuje nowy oraz FSImage na węźle głównym
  - Log transakcyjny węzła głównego jest czyszczony

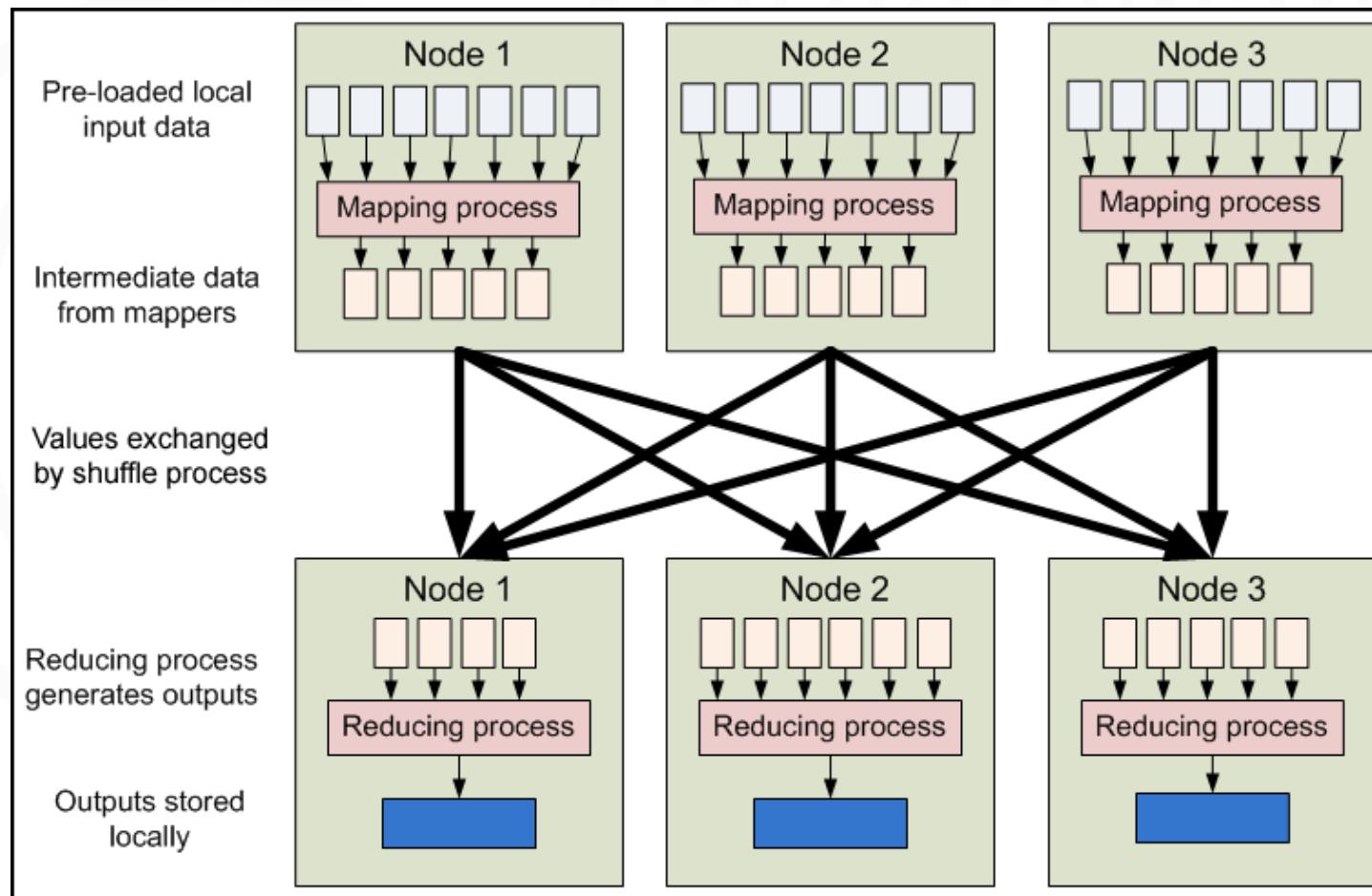
# Interfejs użytkownika

- Wiersz poleceń HDFS dla użytkownika:
  - hadoop dfs -mkdir /foodir
  - hadoop dfs -cat /foodir/plik.txt
  - hadoop dfs -rm /foodir/plik.txt
- Wiersz poleceń HDFS dla administratora
  - hadoop dfsadmin -report
  - hadoop dfsadmin -decommission datanodename
- Interfejs Web
  - <http://host:port/dfshealth.jsp>

# Co to MapReduce?

- MapReduce to model programowania dla efektywnego przetwarzania rozproszonego
- Działa jak przetwarzanie potokowe w Unix-e
  - cat input | grep | sort | uniq -c | cat > output
  - Input | Map | Shuffle & Sort | Reduce | Output
- Wydajność pochodzi z:
  - Streaming przez dane, zmniejszając proces poszukiwania
  - przetwarzanie potokowe
- Dobrym rozwiązaniem dla wielu aplikacji
  - Przetwarzanie log-ów
  - Budowanie indeksu Web

# MapReduce – przepływ danych



źródło: <http://developer.yahoo.com>

# MapReduce - Właściwości

- Podzielone na mniejsze zadania Map i Reduce
  - Udoskonalone równoważenie obciążenia
  - Szybszy powrót do stanu normalnego z awarii
- Automatyczne ponowne wykonania w przypadku awarii
  - W klastrze, niektóre węzły są zawsze wolne lub ułomne
  - Framework ponowne wykonana nieudane zadania
- Optymalizacja lokalizacji
  - Przy BigData, przepustowość danych jest problemem
  - MapReduce + HDFS jest bardzo skuteczne
  - MapReduce pyta HDFS jaka jest lokalizacja danych wejściowych
  - Zadania Map planowane blisko wejścia, jeśli to możliwe



## Programowanie aplikacji w chmurze obliczeniowej

## Elementy aplikacji w chmurze komputerowej

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni



**Fundusze  
Europejskie**  
Wiedza Edukacja Rozwój

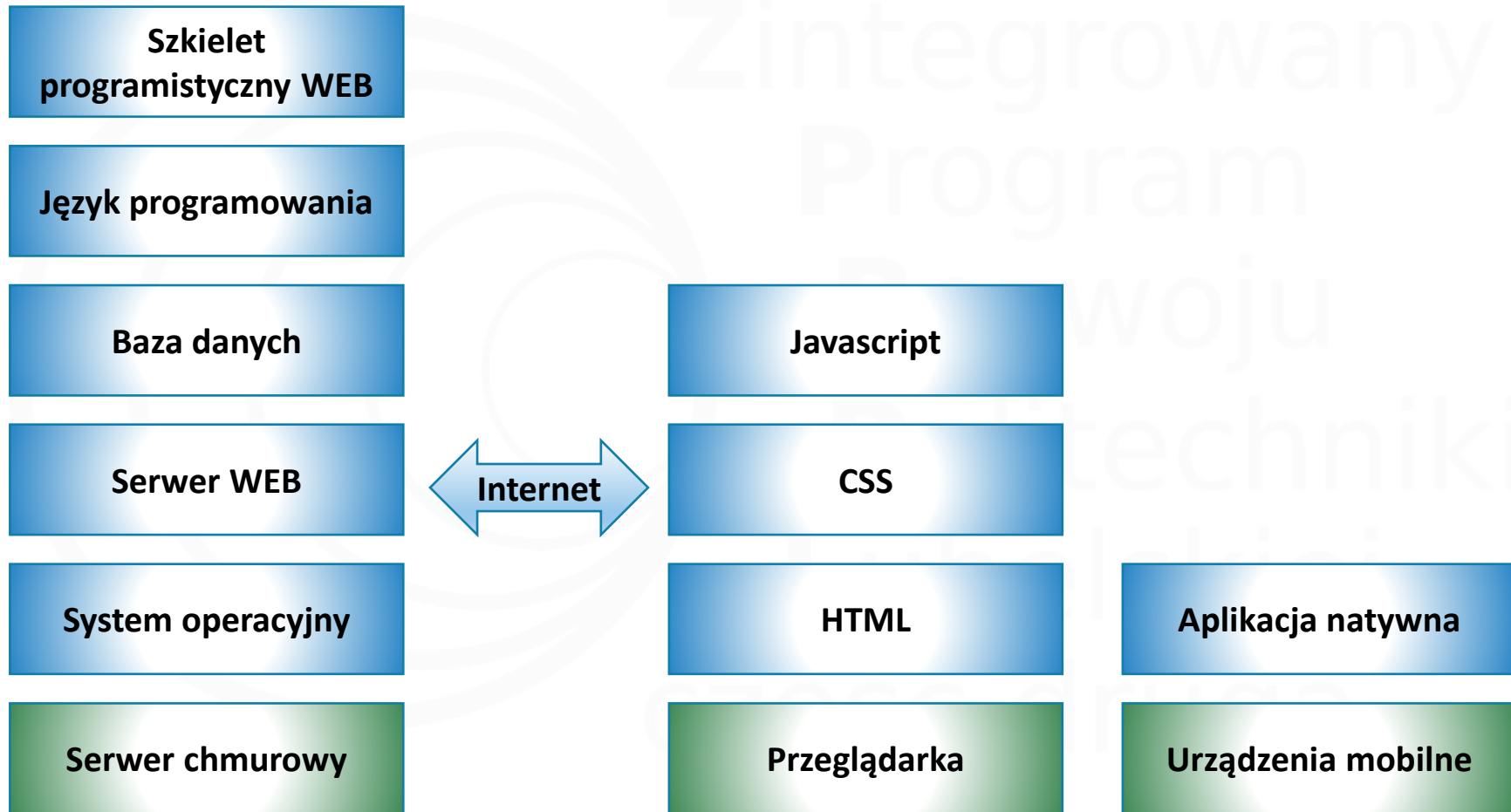


**Rzeczpospolita  
Polska**

**Unia Europejska**  
Europejski Fundusz Społeczny



# Struktura aplikacji chmurowej



# DevOps a aplikacje chmurowe

## (1)

- Podstawowe narzędzia do zarządzania konfiguracją:
  - Chef: używa Ruby, języka specyficznego dla domeny (DSL), do pisania skryptów konfiguracji systemu. Agenci są instalowani za pomocą narzędzia „nóż” z protokołem SSH. Zarządzane węzły uwierzytelniają się za pomocą certyfikatów.
  - Puppet: Służy do zarządzania orkiestracją centrum danych. Działa z wieloma systemami operacyjnymi i zapewnia narzędzia wsparcia operacyjnego dla głównych systemów operacyjnych. Instalacja wymaga serwera głównego i agentów klienta w każdym systemie zarządzanym.

# DevOps a aplikacje chmurowe (2)

- Podstawowe narzędzia do zarządzania konfiguracją:
  - Ansible: Ansible nie wymaga instalacji agenta węzła do zarządzania konfiguracjami. Używa Pythona z instalacją przez klon repozytorium GIT. Wszystkie funkcje używają SSH. Zarządzanie węzłami Ansible wymaga dołączenia autoryzowanych kluczy SSH do każdego węzła.
  - Salt: narzędzie interfejsu wiersza poleceń (CLI), które wykorzystuje metodę wypychania do komunikacji z klientem. Jest instalowany w systemie Git lub systemie zarządzania pakietami. Sól komunikuje się przez ogólny SSH. Zawiera również asynchroniczny serwer plików, który przyspiesza udostępnianie plików. Python lub PyDSL są używane do pisania niestandardowych modułów.

# Mikroserwisy i architektury bezserwerowe

- Jedną ze znaczących zmian w technikach projektowania aplikacji jest styl architektoniczny mikroserwisów.
- Aplikacje są zorganizowane jako zbiór luźno powiązanych usług, które łączą się w celu wdrożenia możliwości biznesowych.
- Architektura mikrousług służy do obsługi ciągłego dostarczania/wdrażania dużych, złożonych aplikacji.
- Umożliwia także organizacji rozwijanie stosu technologii.
- Innym ważnym nowym podejściem jest wyłączne korzystanie z usług stron trzecich, które dostarczają efemeryczne kontenery przy użyciu modelu udostępniania infrastruktury just-in-time, zwanego przetwarzaniem bezserwerowym.
  - Allokacja zasobów maszynowych.
  - Ceny są ustalane na podstawie rzeczywistej ilości zasobów zużywanych

# Plan migracji aplikacji

- Migracje aplikacji zazwyczaj przechodzą przez cztery odrębne fazy:
  - Ocena organizacyjna
  - Definicja i projekt rozwiązania
  - Migracja aplikacji
  - Aplikacje narzędziowe

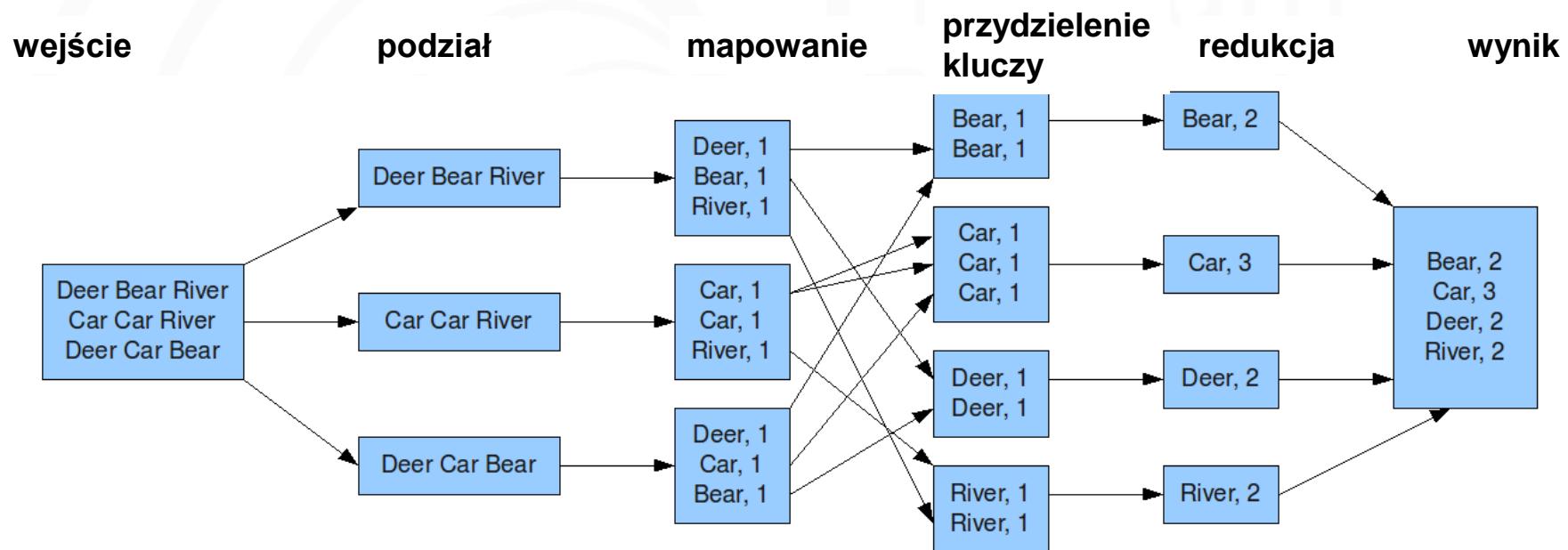
# Procesy przejściowe w migracji

- Aplikacje migrujące są zazwyczaj przeznaczone do jednego z czterech procesów przejściowych:
  - Lift and shift: wymagana infrastruktura jest odbudowywana przy użyciu odpowiednich usług Cloud Solution Providers, a aplikacja jest przenoszona bez żadnych modyfikacji.
  - Refactor: Aplikacje zaprojektowane do działania w niestandardowej infrastrukturze są modyfikowane w celu wykorzystania dostępnych usług w chmurze przed migracją.
  - Rebuild: aplikacje, które są nadal wymagane przez organizację, ale nie można ich modyfikować, aby korzystać z dostępnych usług w chmurze. Te aplikacje są przeprojektowywane i przebudowywane przed przeniesieniem procesu do chmury.
  - Retire: aplikacje, które nie są już operacyjnie lub ekonomicznie opłacalne dla organizacji. Powiązane procesy są albo eliminowane, albo zastępowane dostępnym SaaS.

# Przykład - zliczanie słów

- Mapper
  - Wejście: wartości: wiersze tekstu wejściowego
  - Wyjście: klucz: słowo, wartość: 1
- Reducer
  - Wejście: klucz: słowo, wartość: liczba wystąpień
  - Wyjście: klucz: słowo, wartość: suma
- Uruchomienie programu
  - Definiuje zadanie
  - Wysyła zadanie do chmury

# Zliczanie słów – przepływ danych





## Programowanie aplikacji w chmurze obliczeniowej

### Wykorzystanie języka Java do programowania w chmurze komputerowej

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni



**Fundusze  
Europejskie**  
Wiedza Edukacja Rozwój



**Rzeczpospolita  
Polska**

**Unia Europejska**  
Europejski Fundusz Społeczny



# Zliczanie słów - Mapper

```
public static class Map extends MapReduceBase implements
Mapper<LongWritable,Text,Text,IntWritable> {
private static final IntWritable one = new IntWritable(1);
private Text word = new Text();

public static void map(LongWritable key, Text value,
OutputCollector<Text,IntWritable> output, Reporter reporter)
throws IOException {
    String line = value.toString();
    StringTokenizer tokenizer = new StringTokenizer(line);
    while(tokenizer.hasNext()) {
        word.set(tokenizer.nextToken());
        output.collect(word,one);
    }
}
}
```

# Zliczanie słów - Reducer

```
public static class Reduce extends MapReduceBase implements  
    Reducer<Text, IntWritable, Text, IntWritable> {  
    public static void map(Text key, Iterator<IntWritable> values,  
        OutputCollector<Text, IntWritable> output, Reporter reporter)  
        throws IOException {  
        int sum = 0;  
        while(values.hasNext()) {  
            sum += values.next().get();  
        }  
        output.collect(key, new IntWritable(sum));  
    }  
}
```

# Przykład zliczanie słów

- Zadania są kontrolowane przez konfiguracje JobConfs
- Konfiguracje JobConfs są mapami nazw atrybutów do wartości ciągów znaków
- Framework definiuje atrybuty do kontrolowania sposobu wykonania zadania
  - `conf.set("mapred.job.name", "MyApp");`
- Aplikacje mogą dodać dowolne wartości do JobConf
  - `conf.set("my.string", "foo");`
  - `conf.set("my.integer", 12);`
- JobConf jest dostępny dla wszystkich zadań

# Wszystko razem

- Opracować program dla uruchomienia aplikacji
- Program uruchomieniowy definiuje:
  - Użycie Mapera i Reducera
  - Wyjściowy klucz i typ wartości (typy wejściowe są wywnioskować z wejścia (InputFormat))
  - Lokalizacje dla wejścia i wyjścia
- Program uruchomieniowy następnie wysyła zadanie i zwykle czeka na jego zakończenie

# Wszystko razem

```
JobConf conf = new JobConf(WordCount.class);
conf.setJobName("wordcount");

conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(IntWritable.class);

conf.setMapperClass(Map.class);
conf.setCombinerClass(Reduce.class);
conf.setReducer(Reduce.class);

conf.setInputFormat(TextInputFormat.class);
Conf.setOutputFormat(TextOutputFormat.class);

FileInputFormat.setInputPaths(conf, new Path(args[0]));
FileOutputFormat.setOutputPath(conf, new Path(args[1]));

JobClient.runJob(conf);
```

# Formaty wejścia i wyjścia

- Mechanizm MapReduce może zaspecyfikować jak mają być czytane dane wejściowe poprzez podanie w interfejsie InputFormat
- Mechanizm MapReduce może zaspecyfikować jak mają być zapisywane dane wyjściowe poprzez podanie w interfejsie OutputFormat
- Domyślnie interfejsy wskazują na TextInputFormat oraz TextOutputFormat, które przetwarzają wierszowe dane tekstowe
- Innym popularnym wyborem jest użycie formatów SequenceFileInputFormat oraz SequenceFileOutputFormat dla danych binarnych
- Mechanizmy te są zorientowane na pliki ale nie ma konieczności używania plików.

# Interfejs InputFormat

- `org.apache.hadoop.mapred`  
`Interface InputFormat<K, V>`
- **Zależne podinterfejsy:**  
`ComposableInputFormat<K, V>`
- **Wszystkie klasy implementujące:**  
`CombineFileInputFormat,`  
`CombineSequenceFileInputFormat,`  
`CombineTextInputFormat,` `CompositeInputFormat,`  
`DBInputFormat,` `FileInputFormat,`  
`FixedLengthInputFormat,`  
`KeyValueTextInputFormat,`  
`MultifileInputFormat,` `NLineInputFormat,`  
`Parser.Node,` `SequenceFileAsBinaryInputFormat,`  
`SequenceFileAsTextInputFormat,`  
`SequenceFileInputFilter,`  
`SequenceFileInputFormat,` `TextInputFormat`

# Interfejs InputFormat

- Szkielet aplikacji Map-Reduce wykorzystuje InputFormat aby:
  - Zwaliować specyfikację wejściową zadania.
  - Podzielić pliki wejściowe na logiczne części InputSplits, które są przypisane do indywidualnych maperów.
  - Zapewnić realizację implementacji RecordReader, aby zebrać rekordy wejściowe z InputSplit do przetworzenia przez Mapper.

# Interfejs InputFormat

- Rozmiar bloku danych dla klasy FileSystem jest traktowany jako górnna granica dla każdej porcji wejściowej. Dolna wartość wejściowego bloku danych może być zdefiniowana przez **mapreduce.input.fileinputformat.split.minsize**
- Jasno widać, że podział danych wejściowych ze względu na rozmiar nie jest wystarczający, ponieważ granice zakresu rekordów mają być przestrzegane.
- W takich przypadkach należy zastosować klasę **RecordReader**, na której leży odpowiedzialność za przestrzeganie granic zakresu rekordów i przedstawianie zorientowanego na rekordy widoku **InputSplit** dla danego zadania.

# Interfejs InputFormat

Modyfikator i typ

RecordReader<K,V>

InputSplit[]

Metoda

getRecordReader(InputSplit split,  
JobConf job, Reporter reporter)  
Pobiera RecordReader dla danego  
InputSplit.

getSplits(JobConf job,  
int numSplits)

Logicznie dzieli zestaw plików  
wejściowych do pracy.

# Mapowania i redukcje

- Mapowania

- Zwykle tyle ile wynosi liczba przetwarzanych bloków HDFS, jest to wartość domyślna
- lub - liczbę map można określić jako parametr
- Liczba map może być również kontrolowana poprzez specyfikację minimalnego rozmiaru podziału.
- Aktualne rozmiary wejść do mapera są liczone według:
  - $\max(\min(\text{block\_size}, \text{data}/\#\text{maps}), \text{min\_split\_size})$

- Redukcje

- Dopóki ilość przetwarzanych danych jest mała
  - $0.95 * \text{num\_nodes} * \text{mapred.tasktracker.tasks.maximum}$

# Użyteczne narzędzia

- Partitioners - partycjonery
- Combiners - sumatory
- Compression - kompresja
- Counters - liczniki
- Speculation – spekulatywność
- Zero Reduces – redukcja zer
- Distributed File Cache – rozproszony bufor plików
- Tool – klasa Tool (narzędzie)

# Partitioners - partycjonery

- Partycjonery to aplikacje, które definiują jak klucze są przypisane do reduktorów
- Domyślnie partycjonowanie rozprzestrzenia klucze równomiernie, ale losowo
  - **Uses `key.hashCode() % num_reduces`**
- Niestandardowe partycjonowanie jest często konieczne, na przykład w celu określenia całkowitej wartości wyjścia
  - powinno implementować interfejs Partycjonera
  - być ustawiane przez  
**`conf.setPartitionerClass(MyPart.class)`**
  - aby uzyskać całkowity porządek, powinno spróbować klucze wyjściowe mapowania i wybrać wartości tak aby podzielić klucze w przybliżeniu na równe porcje i użyć ich w partycjonerze

# Combiners - Sumatory

- Kiedy proces mapowania daje wiele powtarzalnych kluczy to:
  - dobrze jest przeprowadzić lokalną agregację uzależnioną od mapy
  - wykonać proces przez określonego Sumatora
  - celem jest zmniejszenie ilości danych przejściowych
  - sumatory mają ten sam interfejs co Reduktory i są często tą samą klasą
  - sumatory nie mogą dawać wyników cząstkowych ponieważ są wielokrotnie pośrednio uruchamiane
  - w przykładzie WordCount:
    - `conf.setCombinerClass(Reduce.class);`

# Kompresja wyjścia

- Kompresja wyjścia i danych pośrednich często przynieść ogromne zyski wydajności
  - może być zaspecyfikowana przez plik konfiguracyjny lub ustawiona w programie
  - Należy ustawić parametr **mapred.output.compress=true** aby skompresować wyjście zadania
  - Należy ustawić **mapred.compress.map.output=true** aby skompresować wyjście mapera

# Kompresja wyjścia

- Typy kompresji  
**mapred(.map)? .output.compression.type**
  - „block” - grupa kluczy i wartości jest kompresowana razem
  - „record” – każda wartość jest kompresowana indywidualnie
  - Zazwyczaj najlepsza jest kompresja bloków
- Kompresja – typy algorytmów kompresujących  
**mapred(.map)? .output.compression.codec**
  - domyślnie (zlib) – wolniejsze, ale lepsza kompresja
  - LZO - szybsze, gorsza kompresja

# Counters - liczniki

- Często aplikacje MapReduce mają policzalne zdarzenia
- Na przykład środowisko zlicza rekordy wejściowe i wyjściowe Mapera i Sumatora
- Aby zdefiniować liczniki użytkownika:
  - `static enum Counter {EVENT1, EVENT2};`
  - `reporter.incrCounter(Counter.EVENT1, 1);`
- Definicja przyjaznych nazw w pliku  
**MyClass\_Counter.properties**

CounterGroupName=MyCounters

EVENT1.name=Event 1

EVENT2.name=Event 2

# Spekulatywne wykonanie

- Zdolność mikroprocesorów przetwarzających instrukcje potokowo do wykonywania instrukcji znajdujących się za skokiem warunkowym, co do którego jeszcze nie wiadomo, czy nastąpi.
- Ostatecznie wyniki wyliczone z wyprzedzeniem zostaną albo uwzględnione, albo odrzucone, zależnie od tego, czy skok się wykona.

# Spekulatywne wykonanie

- Mechanizm redukcji opóźnień jest heurystyczny.
- Procesor próbuje „zgadnąć” przebieg wykonania programu pobiera i wykonuje kolejne rozkazy.
- Procesor nie wie, czy „zgadywanie” dało właściwy efekt aż do czasu wykonania instrukcji skoku.

# Wykonanie spekulatywne - MapReduce

- Szkielet programowy MapReduce może uruchomić wiele instancji wolnych zadań
  - Są użyte wyniki otrzymane od instancji najwcześniej zwracającej dane
  - Kontrola odbywa się przez zmienną **mapred.speculative.execution**
  - Może dramatycznie zwiększyć opóźnienia w wykonaniu się zadań chmurowych

# Redukcja zer

- Często istnieje potrzeba uruchomienia filtru dla danych wejściowych
  - Tam gdzie nie wymagane jest sortowanie lub tasowanie wymagane przez zadanie
  - Ustawienie liczby reduktorów równej 0
  - Wyniki wyjściowe wędrują od razu do wyjścia OutputFormat na dysku

# Rozproszony bufor plików

- Czasami zachodzi potrzeba wykonać kopię danych na lokalnym komputerze
  - Pobranie 1GB danych dla każdego Mapera jest kosztowne
- W takim przypadku należy zdefiniować listę plików do pobrania w pliku konfiguracyjnym JobConf
- Pliki są pobierane raz na każdym komputerze
- Należy dodać do programu uruchomieniowego:
  - `DistributedCache.addCacheFile(new URI("hdfs://adres:8020/plik"), conf);`
- Do zadania dodać:
  - `Path[] files = DistributedCache.getLocalCacheFiles(conf);`

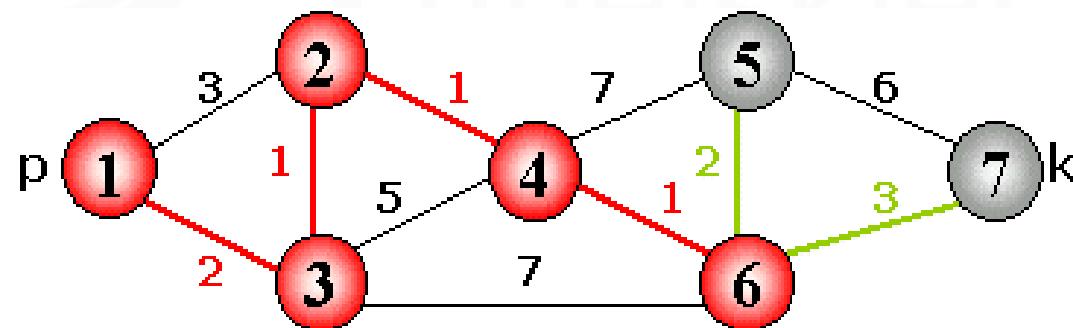
# Klasa Tool

- Wspiera obsługę podstawowych opcji wiersza polecenia
  - **-conf plik1** – załadowuje konfiguracje z pliku o nazwie plik1
  - **-D prop=value** - definiuje pojedynczy parametr i przypisuje mu wartość
- Struktura klasy:

```
public class MyApp extends Configured implements Tool {  
    public static void main(String[] args) throws Exception {  
        System.exit(ToolRunner.run(new Configuration(),  
            new MyApp(), args));  
    }  
    public int run(String[] args) throws Exception {  
        .... getConf() ....  
    }  
}
```

# Znalezienie najkrótszej ścieżki

- Popularnym zadaniem jest znalezienie drogi od węzła startowego do jednego lub więcej węzłów docelowych
- Na pojedynczej maszynie wykorzystany do rozwiązywania jest algorytm Dijkstry
- Czy można użyć przeszukiwania wszerz (breadth-first search) aby znaleźć najkrótszą ścieżkę z wykorzystaniem MapReduce?





# Przeszukiwanie wszerz

- Przechodzenie grafu rozpoczyna się od zadanego wierzchołka  $s$  i polega na odwiedzeniu wszystkich osiągalnych z niego wierzchołków.
- Wynikiem działania algorytmu jest drzewo przeszukiwania wszerz o korzeniu w  $s$ , zawierające wszystkie wierzchołki osiągalne z  $s$ .
- Do każdego z tych wierzchołków prowadzi dokładnie jedna ścieżka z  $s$ , która jest jednocześnie najkrótszą ścieżką w grafie wejściowym.

# Znalezienie najkrótszej ścieżki: rozwiązanie intuicyjne

- Możemy zdefiniować rozwiązanie tego problemu indukcyjnie
  - $DystansDo(węzełStarowy) = 0$
  - Dla wszystkich n węzłów bezpośrednio dostępnych z węzłaStartowego  $DystansDo(n) = 1$
  - Dla wszystkich n węzłów dostępnych z innego zestawu węzłów S mamy:
    - $DystansDo(n) = 1 + \min(DystansDo(m), m \in S)$

# Od intuicji do algorytmu

- Zadanie mapowania otrzymuje numer węzła  $n$  jako klucz, a  $(D, \text{punkty}-\text{do})$  jako jego wartości
  - $D$  to dystans do węzła z punktu startowego
  - $\text{punkty}-\text{do}$  jest to lista węzłów osiągalnych z  $n$
- $\forall p \in \text{punkty}-\text{do}$ , emituj  $(p, D+1)$   
(dla każdego  $p$  należącego do punkty-do)
- Zadania redukcji gromadzą możliwe odległości dla danego  $p$  i wybierają najmniejszą

# Rozwiążanie

- Zadanie MapReduce może osiągnąć granicę grafu w jednym przeskoku
- Aby przeprowadzić całe przeszukiwanie wszerz komponent nie będący w MapReduce przekazuje wyjście tego kroku do zadania MapReduce do kolejnej iteracji
  - Problem: gdzie podzieliła się lista punkty-do?
  - rozwiązanie: mapper emituje również (n, punkty-do)

# Zakończenie wykonania

- Algorytm ten staruje z jednego węzła grafu
- Kolejne iteracje zawierają wiele więcej węzłów grafu jako dane wejściowe
- Czy algorytm się kiedyś kończy?
  - Tak, ostatecznie trasy między węzłami nie będą odkrywane i nie zostanie znaleziona lepsza odległość. Gdy odległość jest taka sama, możemy przestać.
  - Mapper powinien emitować  $(n, D)$  aby się upewnić, że bieżący dystans jest przekazywany do reducera



## Programowanie aplikacji w chmurze obliczeniowej

## Usługi Pig i Hive jako elementy chmury komputerowej

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni



**Fundusze  
Europejskie**  
Wiedza Edukacja Rozwój



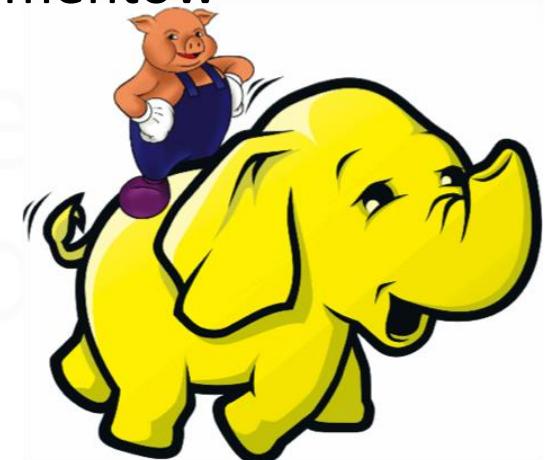
**Rzeczpospolita  
Polska**

**Unia Europejska**  
Europejski Fundusz Społeczny



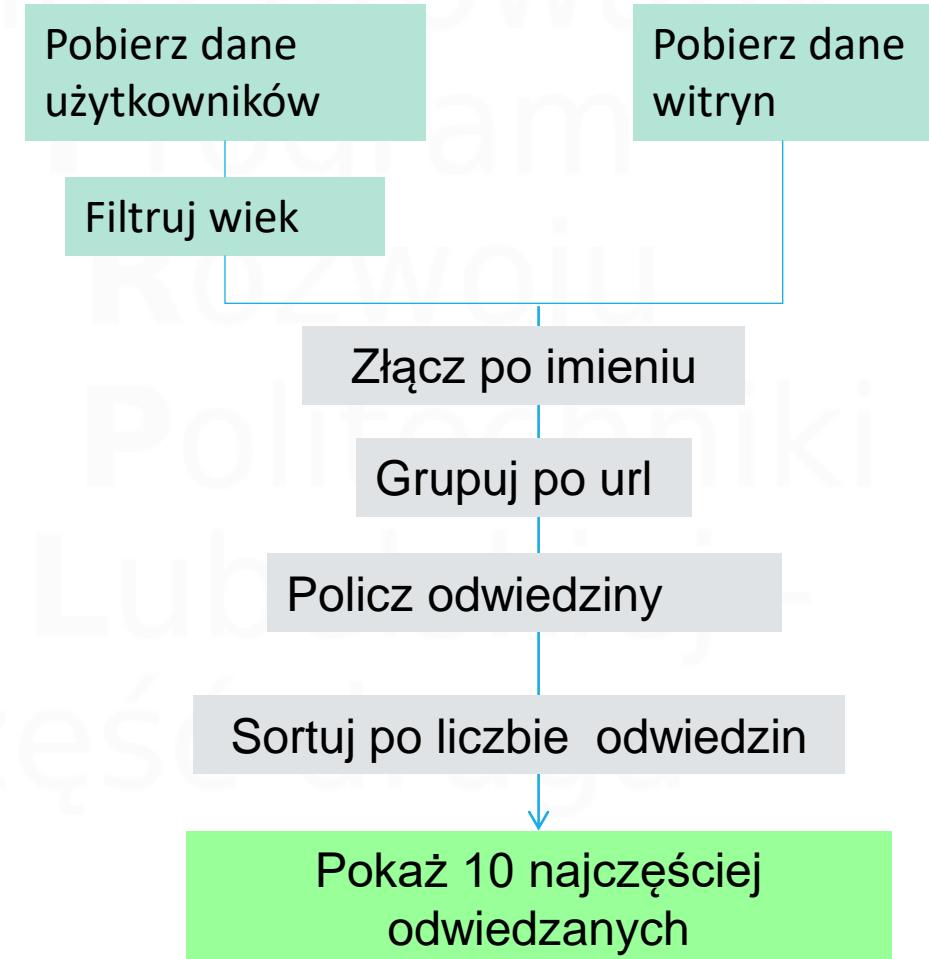
# Pig

- Początki Yahoo! Research 2006
- Około 30% zadań uruchamianych w Yahoo! to Pig Latin
- Własności
  - Wyraża sekwencje w zadaniach MapReduce
  - Model danych: zagnieżdżone „worki” elementów
  - Dostarcza operatorów relacyjnych (SQL) takich jak (JOIN, GROUP BY, UNION, EXCEPT, ipt)
  - Łatwy dołączenia w funkcjach Java



# Przykładowy problem

- Dane na temat odwiedzin użytkowników są zgromadzone w jednym pliku
- Dane na temat odwiedzanych witryn są zgromadzone w innym pliku
- Znaleźć 10 najczęściej odwiedzanych witryn przez użytkowników we wcześniejszym okresie dorosłości (20-34)



# W MapReduce Java

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.InputFormat;
import org.apache.hadoop.mapred.KeyValueTextInputFormat;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.mapred.SequenceFileInputFormat;
import org.apache.hadoop.mapred.SequenceFileOutputFormat;
import org.apache.hadoop.mapred.TextLineInputFormat;
import org.apache.hadoop.mapred.jobcontrol.Job;
import org.apache.hadoop.mapred.jobcontrol.JobControl;
import org.apache.hadoop.mapred.lib.IdentityMapper;

public class MRExample {
    public static class LoadPages extends MapReduceBase
        implements Mapper<LongWritable, Text, Text, Text> {
        public void map(LongWritable k, Text val,
            OutputCollector<Text, Text> oc,
            Reporter reporter) throws IOException {
            // Pull the key out
            String line = val.toString();
            int firstComma = line.indexOf(',');
            String key = line.substring(0, firstComma);
            String value = line.substring(firstComma + 1);
            Text outKey = new Text(key);
            // Prepend an index to the value so we know which file
            // it came from
            Text outVal = new Text("1" + value);
            oc.collect(outKey, outVal);
        }
    }

    public static class LoadAndFilterUsers extends MapReduceBase
        implements Mapper<LongWritable, Text, Text, Text> {
        public void map(LongWritable k, Text val,
            OutputCollector<Text, Text> oc,
            Reporter reporter) throws IOException {
            // Pull the key out
            String line = val.toString();
            int firstComma = line.indexOf(',');
            String key = line.substring(0, firstComma);
            String value = line.substring(firstComma + 1);
            if (key.length() < 1) return;
            String outKey = new Text(key);
            // Prepend an index to the value so we know which file
            // it came from
            Text outVal = new Text("2" + value);
            oc.collect(outKey, outVal);
        }
    }

    public static class Join extends MapReduceBase
        implements Reducer<Text, Text, Text, Text> {
        public void reduce(Text key,
            Iterator<Text> iter,
            OutputCollector<Text, Text> oc,
            Reporter reporter) throws IOException {
            // For each value, figure out which file it's from and
            store it accordingly.
            List<String> first = new ArrayList<String>();
            List<String> second = new ArrayList<String>();

            while (iter.hasNext()) {
                Text t = iter.next();
                String value = t.toString();
                if (value.charAt(0) == '1')
                    first.addValue(substring(1));
                else second.addValue(value.substring(1));
            }
        }
    }

    public static void main(String[] args) throws IOException {
        JobConf lp = new JobConf(MRExample.class);
        lp.setJobName("Load Pages");
        lp.setInputFormat(TextInputFormat.class);
        lp.setOutputFormat(TextOutputFormat.class);
        Path lpath = new Path("/user/gates/tmp/indexed_pages");
        lp.setNumReduceTasks(0);
        Job loadPages = new Job(lp);

        JobConf lfu = new JobConf(MRExample.class);
        lfu.setJobName("Load and Filter Users");
        lfu.setInputFormat(TextInputFormat.class);
        lfu.setOutputClass(Text.class);
        lfu.setMapperClass(LoadAndFilterUsers.class);
        lfu.setOutputFormat(TextOutputFormat.class);
        Path lfpath = new Path("/user/gates/tmp/filterd_users");
        lfu.setNumReduceTasks(0);
        Job loadUsers = new Job(lfu);

        JobConf join = new JobConf(MRExample.class);
        join.setMapperClass(KeyValueTextInputFormat.class);
        join.setMapperClass(TextOutputFormat.class);
        join.setOutputKeyClass(Text.class);
        join.setOutputValueClass(Text.class);
        join.setMapperClass(Join.class);
        join.setCombinerClass(JoinCombiner.class);
        FileInputFormat.addInputPath(join, new Path("/user/gates/tmp/indexed_pages"));
        FileOutputFormat.setOutputPath(join, new Path("/user/gates/tmp/joined"));
        join.setNumReduceTasks(50);
        Job joinJob = new Job(join);
        joinJob.addDependingJob(loadPages);
        joinJob.addDependingJob(loadUsers);

        JobConf group = new JobConf(MRExample.class);
        group.setJobName("Group URLs");
        group.setInputFormat(KeyValueTextInputFormat.class);
        group.setOutputKeyClass(Text.class);
        group.setOutputFormat(TextOutputFormat.class);
        group.setMapperClass(LoadJoined.class);
        group.setCombinerClass(ReduceUrls.class);
        group.setReducerClass(ReduceUrls.class);
        FileInputFormat.addInputPath(group, new Path("/user/gates/tmp/joined"));
        FileOutputFormat.setOutputPath(group, new Path("/user/gates/tmp/grouped"));
        group.setNumReduceTasks(50);
        Job groupJob = new Job(group);
        groupJob.addDependingJob(joinJob);

        JobConf top100 = new JobConf(MRExample.class);
        top100.setJobName("Top 10 sites");
        top100.setInputFormat(SequenceFileInputFormat.class);
        top100.setOutputFormat(TextOutputFormat.class);
        top100.setOutputValueClass(Text.class);
        top100.setOutputFormat(TextOutputFormat.class);
        top100.setMapperClass(LoadClicks.class);
        top100.setCombinerClass(LimitClicks.class);
        top100.setReducerClass(LimitClicks.class);
        FileInputFormat.addInputPath(top100, new Path("/user/gates/tmp/grouped"));
        FileOutputFormat.setOutputPath(top100, new Path("/user/gates/tmp/top100"));
        top100.setNumReduceTasks(1);
        Job limit = new Job(top100);
        limit.addDependingJob(groupJob);

        JobControl jc = new JobControl("Find top 100 sites for users
        18 to 25");
        jc.addJob(loadPages);
        jc.addJob(loadUsers);
        jc.addJob(joinJob);
        jc.addJob(groupJob);
        jc.addJob(limit);
        jc.run();
    }
}
```

# W Pig Latin

```
Uzytkownicy = load 'users' as (imie, wiek);  
Filruj = filter Users by wiek >= 20 and wiek <= 34;  
Witryny = load 'pages' as (uzyt, url);  
Zlacz = join Filruj by imie, Witryny by uzyt;  
Grupuj = group Zlacz by url;  
Sumuj = foreach Grupuj generate group,  
           count(Zlacz) as odwiedziny;  
Sortuj = order Sumuj by odwiedziny desc;  
Top10 = limit Sortuj 10;  
store Top10 into 'top10witryn';
```

# Łatwość translacji

Pobierz dane użytkowników

Pobierz dane witryn

Filtruj wiek

Złącz po imieniu

Grupuj po url

Policz odwiedziny

Sortuj po liczbie odwiedzin

Pokaż 10 najczęściej odwiedzanych

Użytkownicy = **load** ...

Filruj = **filter** ...

Witryny = **load** ...

Zlacz = **join** ...

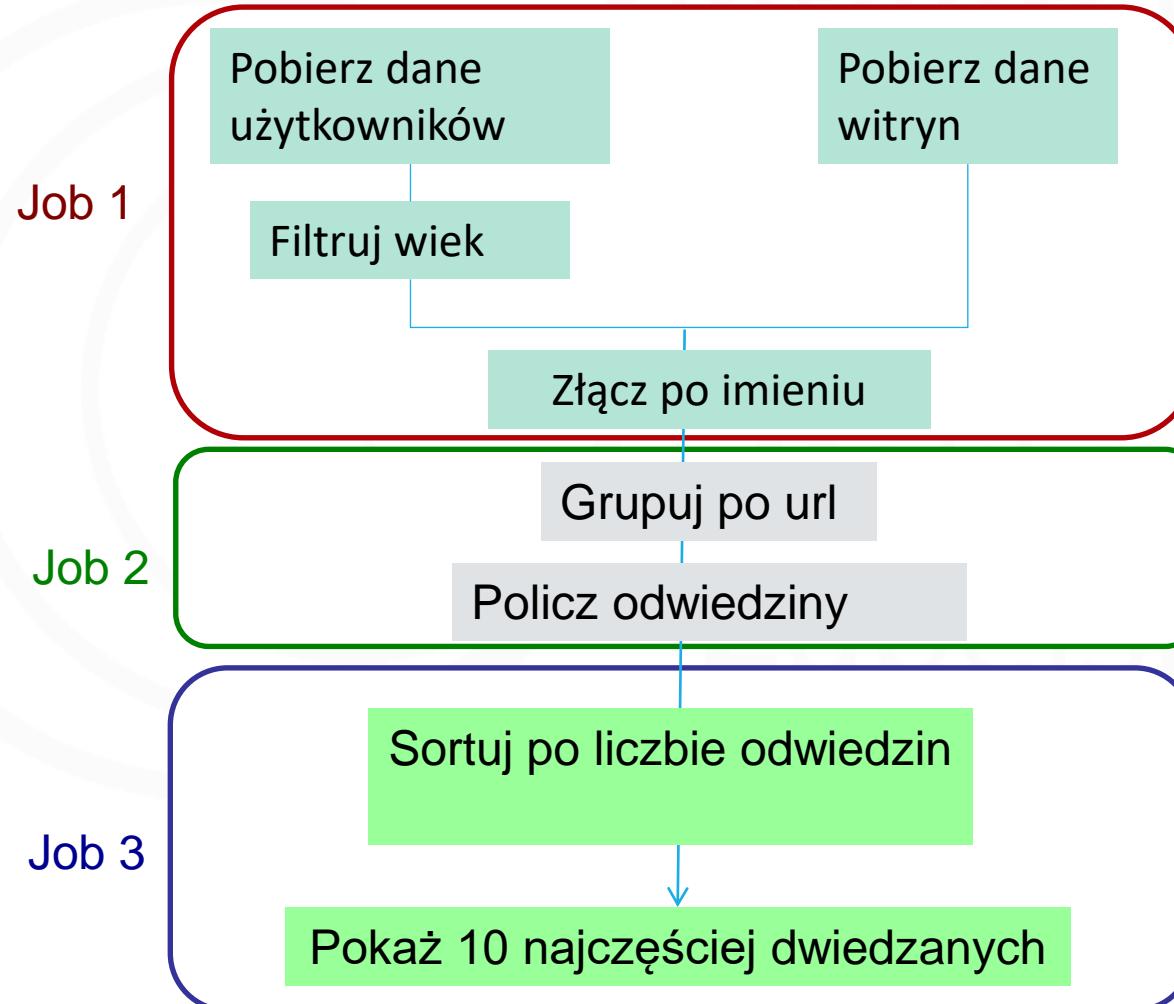
Grupuj = **group** ...

Sumuj = **count**...

Sortuj = **order**...;

Top10 = **limit** Sortuj 10;

# Łatwość translacji



# Apache Pig

- Apache Pig jest platformą do analizy dużych zbiorów danych.
- Pig składa się z języka wysokiego poziomu do wyrażania programów analizy danych, w połączeniu z infrastrukturą do uruchamiania tych programów.

# Apache Pig

- Warstwa infrastruktury Pig składa się z:
  - kompilatora, który generuje sekwencje programów Map-Reduce
  - warstwa języka Pig składa się obecnie z języka tekstu zwanego Pig Latin, który ma następujące główne właściwości:
    - Łatwość programowania.
    - Łatwo osiągnąć równoległe wykonywanie prostych zadań przetwarzających dane
    - Kompleksowe zadania składające się z wielu powiązanych ze sobą transformacji danych są wyraźnie zakodowane jako sekwencje przepływu danych, dzięki czemu są łatwe do napisania, zrozumienia i utrzymania.

# Pig Latin

- Możliwości optymalizacji:
  - Sposób, w który są wpisane zadania umożliwia systemowi automatyczną optymalizację ich wykonania, co pozwala użytkownikowi na skupienie się na semantyce zamiast wydajności.
- Rozszerzalność:
  - Użytkownicy mogą tworzyć własne funkcje aby opracować przetwarzanie specjalnego przeznaczenia.

# Uruchomienie Pig

- Aby wpisać polecenia Pig Latin należy:

- używając wiersza poleceń grunt

```
$ pig ... - Connecting to ...
```

```
grunt> dane = load 'data';
```

```
grunt> witryny = ... ;
```

- w trybie lokalnym lub MapReduce

```
$ pig zadanie.pig
```

```
Przetwarzanie wsadowe w trybie lokalnym
```

```
$ pig -x local zadanie.pig
```

- zarówno interaktywnie jak i wsadowo

# Organizacja przepływu programu

- Polecenie LOAD czyta dane z systemu plików
- Szereg poleceń przekształcających przetwarza dane
- Polecenie STORE zapisuje dane do systemu plików
- Polecenie DUMP wyświetla wynik na ekranie

# Interpretacja

- Ogólnie, Pig przetwarza wypowiedzi Pig Latin następująco:
  - Po pierwsze, Pig sprawdza składnię i semantykę wszystkich poleceń
  - Następnie, jeśli Pig napotka w kodzie DUMP lub STORE, Pig wykona wszystkie polecenia

# Interpretacja

```
A = LOAD 'student' USING PigStorage() AS (imie:chararray,  
wiek:int, srednia:float);
```

```
B = FOREACH A GENERATE imie
```

```
DUMP B;
```

```
(Jan)
```

```
(Maria)
```

```
(Krzysztof)
```

- Operator STORE zapisuje dane do pliku

# Przykłady

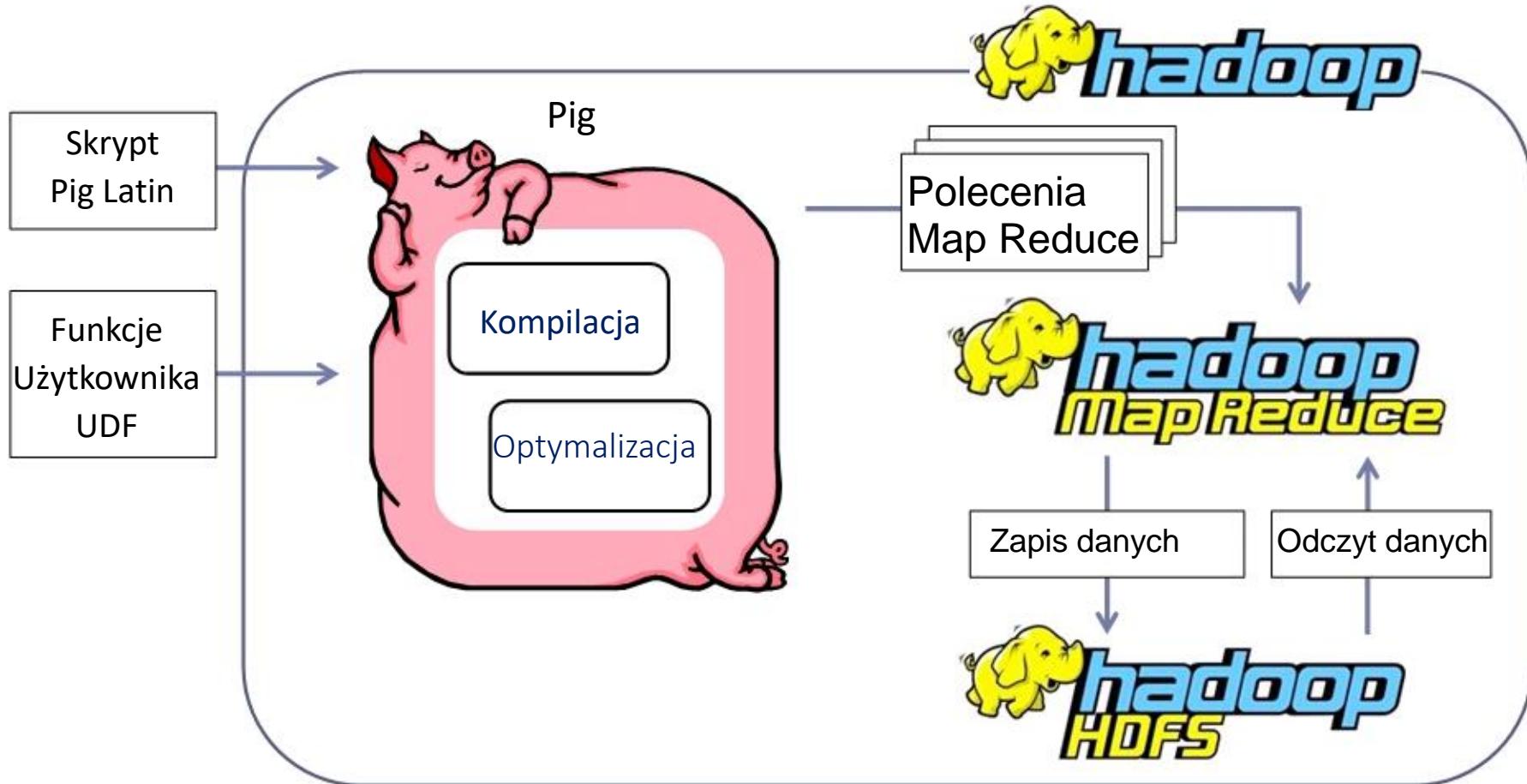
```
punkty1 = LOAD 'input' AS (x, y, z);  
punkty2 = FILTER punkty1 BY x > 6;  
DUMP punkty2;  
punkty3 = FOREACH punkty2 GENERATE y, z;  
STORE punkty3 INTO 'output';
```

```
punkty1 = LOAD 'input' AS (x, y, z);  
punkty2 = FILTER punkty1 BY x > 6;  
STORE punkty2 INTO 'output1';  
punkty3 = FOREACH punkty2 GENERATE y, z;  
STORE punkty3 INTO 'output2'
```

# Pig Latin a SQL

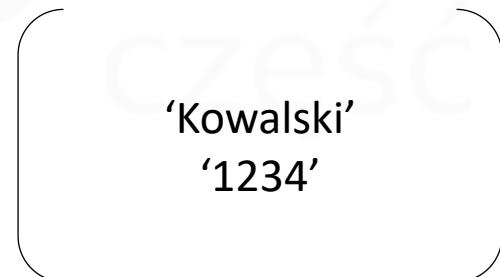
- Table url\_e: (url, kategoria, rangawitryny)
- Znajdź dla każdej dostatecznie dużej kategorii, średnią wartość rangi witryny dla witryn z wysoką rangą
- SQL:
  - ```
SELECT kategoria, AVG(rangawitryny)
  FROM url_e WHERE rangawitryny > 0.4
  GROUP BY kategoria HAVING COUNT(*) > 10^4
```
- Pig Latin:
  - `dobre_url = FILTER url_e BY rangawitryny > 0.4;`
  - `groups = GROUP dobre_urls BY kategoria;`
  - `big_groups = FILTER groups BY COUNT(dobre_url)>10^4;`
  - `output = FOREACH big_groups GENERATE kategoria,
 AVG(gdobre_url.rangawitryny);`

# Zasada działania



# Model danych

- Atom – prosta atomowa wartość (np.: łańcuch lub liczba)
- Krotka - tuple
- Worek - bag
- Mapa - Map



# Model danych

- Atom
- Krotka – sekwencja pól, pola dowolnego typu
- Worek - bag
- Mapa - Map

‘Kowalski’,  $\left\{ \begin{array}{l} (\text{smartfon}, 1) \\ (\text{student}, 2) \end{array} \right\}$ , [‘wiek’->22]

# Model danych

- Atom
- Krotka
- Worek – bag -> kolekcja krotek
  - możliwe duplikaty
  - krotki w worku mogą mieć różną długość i typ pól
- Mapa - Map

{ 'Kowalski', {('smartfon', 1 ), ('student',2) }, ['wiek'->22] }  
{'1234', ('student',3)}

# Model danych

- Atom
- Krotka
- Worek
- Mapa – Map -> kolekcja par klucz-wartość
  - Klucz jest atomowy; wartość może być dowolnego typu

{ 'Kowalski', { ('smartfon',1 ), ('student',2 ) }, ['wiek'->22] ,  
      '1234', ('student',3) }

# Model danych

- Atom
- Krotka
- Worek – bag -> kolekcja krotek
  - możliwe duplikaty
  - krotki w worku mogą mieć różną długość i typ pól
- Mapa - Map

{ 'Kowalski', {('smartfon', 1 ), ('student',2) }, ['wiek'->22] }  
{'1234', ('student',3)}

# Model danych

- Kontrola przepływu danych
- Przykład 1 (mniej efektywny)
  - `spam_url = FILTER url_e BY isSpam(url);`
  - `spammerskie_url = FILTER spam_url BY rangawitryny > 0.8;`
- Przykład 2 (bardziej efektywny)
  - `wysokirk_url = FILTER url_e BY rangawitryny > 0.8;`
  - `spam_url = FILTER wysokirk_url BY isSpam(url);`

# Model danych

- Całkowicie zagnieżdżone
  - Bardziej naturalne dla programistów proceduralnych (użytkownik docelowy) niż normalizacji
  - Dane są często przechowywane na dysku w zagnieżdżony sposób
  - Ułatwia łatwość pisania funkcji zdefiniowanych przez użytkownika
- Nie wymaga schematu

# Model danych

- Funkcje definiowane przez użytkownika (UDF)
  - `spam_url = FILTER url_e BY isSpam(url);`
- Mogą być stosowane w wielu miejscach kodu Pig Latin
- Przydatne do przetwarzania niestandardowych zadań
- Można używać nie atomowych wartości na wejściu i wyjściu
- Obecnie muszą być napisane w języku Java

# Składnia Pig Latin - LOAD

- Wejście zakłada się, że jest workiem (sekwencją krotek)
- Można określić deserializator z „USING”
- Może stanowić schemat z "AS,"
  - nBag = LOAD 'nazwapliku'  
    <USING nazwaFunkcji ()>  
    <AS (Pole1, Pole2, ...)>;
  - Zapytania = LOAD 'log.txt'  
    USING mojaFunkcja()  
    AS (usrID, zapytanie, czas)

# Składnia Pig Latin - FOREACH

- Przetwarza każdą krotkę w worku
- Każde pole może być:
  - Nazwą pola w worku
  - Stałą
  - Wyrażeniem prostym np.:  $x1+x2$
  - Predefiniowaną funkcją (SUM, AVG, COUNT, FLATTEN)
  - Funkcją użytkownika UDF (np: sumOcen(inz, mgr) )
- newBag =  
FOREACH bagName  
GENERATE field1, field2, ...;

# Składnia Pig Latin - FILTER

- Wybór podzbioru krotek w worku
  - **newBag = FILTER bagName BY expression;**
- Wyrażenie używa prostych operatorów porównania (==, !=, <, >, ...) oraz logicznych (AND, NOT, OR)
  - **jakies\_ryby =FILTER ryby BY gatunek != 'plotka';**
- Może używać funkcji UDF
  - **jakies\_ryby =FILTER ryby BY NOT jestPlotka(gatunek);**

**POLITECHNIKA LUBELSKA**

**WYDZIAŁ ELEKTROTECHNIKI I INFORMATYKI**

**INFORMATYKA**



Zintegrowany  
Program  
Rozwoju  
Politechniki  
Lubelskiej -  
część druga

# Hive

## Hurtownia danych



**Fundusze  
Europejskie**  
Wiedza Edukacja Rozwój



**Rzeczpospolita  
Polska**

**Unia Europejska**  
Europejski Fundusz Społeczny



# Środowisko hurtowni danych

## Hive

- Początkowo rozwijane przez Facebook
- Używany do większości prac na Facebooku
- Obecnie Apache Hive używany i rozwijany przez inne przedsiębiorstwa np. Netflix.
- Amazon utrzymuje zmienioną gałąź (fork) oprogramowania w swoich produktach Elastic MapReduce i Amazon Web Services

# Środowisko hurtowni danych

## Hive

- „Relacyjna” baza danych zbudowana na bazie Hadoop
  - Utrzymuje listę schematów tabeli
  - Język zapytań HiveQL bazuje na SQL
  - Można wywoływać skrypty Hadoop Streaming z HiveQL
  - Obsługuje partycjonowanie tabeli, klastry, złożone typy danych, niektóre optymalizacje

# Hive – tworzenie tabeli

```
CREATE TABLE odwiedziny(czasOgl INT, userid BIGINT,  
    data DATE, url_stony STRING, url_skoku STRING,  
    adresip STRING COMMENT 'Adres IP uzytk.' )  
COMMENT 'To jest tablica odwiedzin'  
PARTITIONED BY(data STRING, panstwo STRING)  
STORED AS SEQUENCEFILE;
```

- Partycjonowanie dzieli tabele na oddzielne pliki dla każdej pary (data, panstwo)
- Np: /hive/odwiedziny/data=2015-05-01,panstwo=PL
- /hive/odwiedziny/data=2015-05-01,panstwo=GB

# Zapytanie

- Znajdź wszystkie witryny pochodzące z domeny abc.pl do końca czerwca

```
SELECT odwiedziny.*  
FROM odwiedziny  
WHERE odwiedziny.data >= '2015-06-01'  
AND odwiedziny.data <= '2015-06-30'  
AND odwiedziny.url_skoku like '%abc.pl';
```

- Hive czyta tylko partycję 2015-06-01,\* zamiast całą tabelę

# Agregacje i związki

- Policz użytkowników, którzy wizytowali stronę ze względu na płeć:

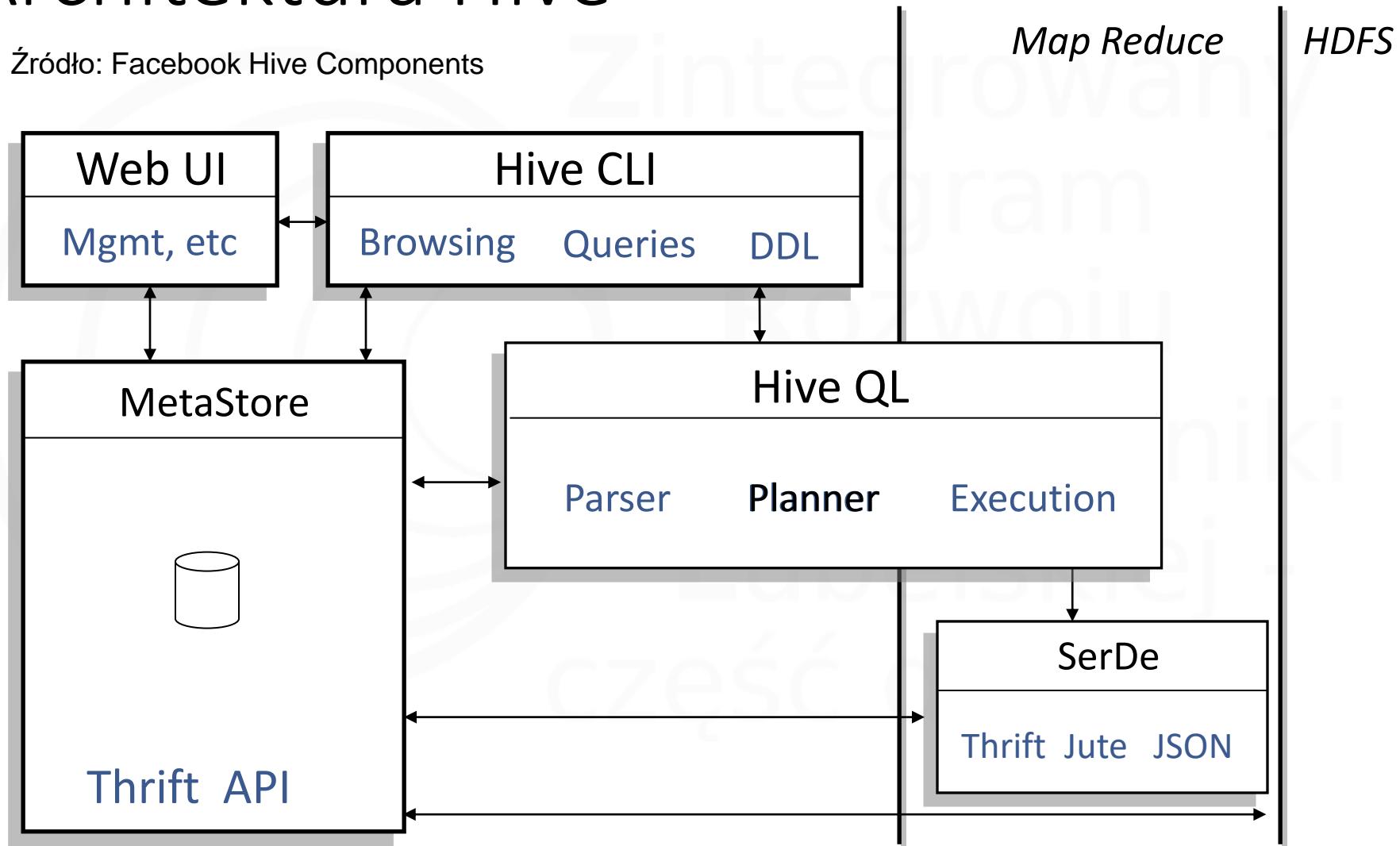
```
SELECT odw.url_strony, uz.plec, COUNT(DISTINCT uz.id)
FROM odwiedziny odw JOIN user uz ON (odw.userid = uz.id)
GROUP BY odw.page_url, uz.plec
WHERE odw.date = '2015-03-01';
```

- Przykładowe wyjście:

url_strony	plec	Count(uz.id)
index.php	M	13,140,320
index.php	F	15,315,450
galeria.php	M	30,34-,482

# Architektura Hive

Źródło: Facebook Hive Components



# Hive QL – złączenia

odwiedziny

idstr	<b>iduz</b>	czas
1	<b>112</b>	8:08:11
2	<b>112</b>	8:09:13
1	<b>223</b>	8:09:22

uzytkownik

<b>iduz</b>	wiek	plec
<b>112</b>	25	female
<b>223</b>	32	male

odw\_uzyt

idstr	wiek
1	25
2	25
1	32

x

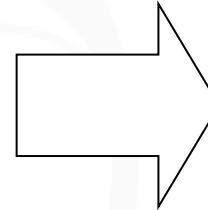
=

- Hive SQL:

```
INSERT INTO TABLE odw_uzyt
SELECT odw.idstr, uz.wiek
FROM odwiedziny odw JOIN uzytkownik uz ON
(odw.iduz = uz.iduz);
```

# Hive QL – klauzula GROUP BY

odw_uzyt	
idstr	wiek
1	25
2	25
1	32
2	25



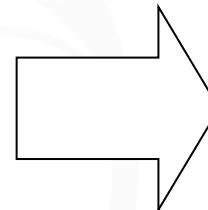
idstr_wiek_suma		
idstr	wiek	Count
1	25	1
2	25	2
1	32	1

- Hive SQL:

```
INSERT INTO TABLE idstr_wiek_suma
SELECT idstr, wiek, count(1)
FROM odw_uzyt
GROUP BY idstr, wiek;
```

# Hive QL – wyrażenie GROUP BY z DISTINCT odwiedziny

idstr	iduz	czas
1	112	8:08:01
2	112	8:08:13
1	223	8:08:14
2	112	8:08:20



rezultat	
idstr	count_distinct_iduz
1	2
2	1

- Hive SQL
  - SELECT idstr, COUNT(DISTINCT iduz)
  - FROM odwiedziny GROUP BY idstr

# Przykłady

```
CREATE EXTERNAL TABLE blokowani_uzyt(
    iduzyt INT,
    blokowany INT,
    bloker INT,
    utworzono BIGINT)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ("hbase.columns.mapping" =
":key,f:blokowany,f:bloker,f:utworzono")
TBLPROPERTIES("hbase.table.name" = "m2h_repl-
userdb.stumble.blokowani_uzyt");
```

- HBase w tym przypadku posiada unikalny klucz wiersza mapy o nazwie :key
- nie wszystkie kolumny w tabeli wymagają mapowania

# Przykłady

```
CREATE EXTERNAL TABLE ratings_hbase(  
    userid INT, created BIGINT, urlid INT,  
    rating INT, topic INT, modified BIGINT)  
STORED BY  
    'org.apache.hadoop.hive.hbase.HBaseStorageHandle  
r'  
WITH SERDEPROPERTIES ("hbase.columns.mapping" =  
    ":key#b@0,:key#b@1,:key#b@2,default:rating#b,def  
ault:topic#b,default:modified#b")  
TBLPROPERTIES ("hbase.table.name" =  
    "ratings_by_userid");
```

- #b oznacza binarny, @ oznacza pozycję w kluczu złożonym

# Hive - typy danych

- Typy kolumn
- Literały
- Wartości NULL
- Typy złożone

# Typy kolumnowe

- Liczby całkowite

Type	Postfix	Example
TINYINT	Y	10Y
SMALLINT	S	10S
INT	-	10
BIGINT	L	10L

# Typy kolumnowe

- Łańcuchy
  - mogą być w (‘ ’) lub (“ ”)

Data Type	Length
VARCHAR	1 to 65355
CHAR	255

# Typy kolumnowe

- **Czas**
- Wsparcie dla tradycyjnego znacznika czasu UNIX z opcjonalną precyzją nanosekundową.
- Wsparcie dla formatu:
  - `java.sql.Timestamp`  
“YYYY-MM-DD HH:MM:SS.fffffffff”
  - “yyyy-mm-dd hh:mm:ss.fffffffff”.

# Typy kolumnowe

- Daty
  - wartości DATE są w formie year/month/day {{YYYY—MM—DD}}.
- Wartości rzeczywiste
- Typ DECIMAL w systemie Hive jest taki sam jak Big Decimal format dla j. Java.
- Jest używany do reprezentowania niezmiennej dowolnej precyzji:
  - Składnia:
  - DECIMAL(precision, scale)
  - np. decimal(10,0)

# Typy UNION

- Unia jest zbiorem różnorodnych typów danych.
- Można utworzyć instancję za pomocą **create union**.
- Przykładowa składnia:
- UNIONTYPE<int, double, array<string>, struct<a:int,b:string>>

# Typy UNION

- {0:1} //int
- {1:2.2} //double
- {2:["trzy", "cztery"]} //array<string>
- {3>{"a":5,"b":„pięć"} } //struct<a:int,b:string>
- Zmiana wartości
- {2:["sześć","siedem"]}
- {3>{"a":8,"b":"osiem"} }

# Literały

- W Hive używane są następujące literały:
- Typy Floating Point
- Typy zmienno-przecinkowe są skomponowane z typu DOUBLE.
- Typy Decimal
- Typ DECIMAL jest wartością zmiennoprzecinkową większą niż DOUBLE. Zakres ten wynosi około  $-10^{-308}$  do  $10^{308}$ .

# Wartości NULL

- Wartości brakujące uzupełniane są poprzez specjalną wartość NULL

# Typy złożone

- Typami złożonymi w Hive są:
- **Arrays** – tablice, używane w analogiczny sposób jak w j. Java
  - Składnia: ARRAY<data\_type>
- **Maps** – mapy są podobne do map Java
  - Składnia: MAP<primitive\_type, data\_type>
- **Structs** – struktury w Hive są podobne do danych złożonych posiadających komentarz
  - Składnia: STRUCT<col\_name : data\_type [COMMENT col\_comment], ...>

# Obróbka danych strukturalnych

- Rodzaj systemu
  - Typy proste
  - Rekursively budowane używając Kompozycji/Map/List
- Generyczny Interfejs Deserializacji (SerDe)
  - Aby rekursively przeglądać schemat
  - Aby rekursively uzyskać dostęp do pól w obiekcie w wierszu
- Rodziny serializacji implementują interfejs
  - SerDe bazujący na Thrift DDL
  - SerDe wykorzystujący rozdzielony tekst
  - Można napisać własny SerDe
- Ewolucja schematu

# MetaMagazyn

- Magazynuje własności Tabel/Partycji:
  - Schemat tabel i bibliotekę SerDe
  - Lokalizację tabel w HDFS
  - Klucze i typy partycjonowania logicznego
  - Inne informacje
- Thrift API
  - Bieżący klient może być napisany w Php (interfejs Web), Python (stare CLI), Java (Query Engine oraz CLI), Perl (Testy)
- Metadane mogą być składowana jako pliki tekstowe a nawet z wykorzystaniem SQL

# Hive CLI

- DDL:
  - Tabele – tworzenie, usuwanie, zmiana nazwy
  - Dodawanie kolumn w tabelach
- Przeglądanie:
  - Pokazywanie nazw tabel (show tables)
  - Własności tabeli (describe table)
  - Wyświetlenie zawartości (cat table )
- Ładowanie danych
- Zapytania

# Interfejs Web UI dla Hive

- HiPal:
  - Interaktywna konstrukcja zapytań SQL z pomocą urządzenia wskazującego
  - Wsparcie dla projekcji, filtrowania, grupowania i łączenia

The screenshot shows the HiPal web interface for querying the Facebook Data Warehouse. The top navigation bar includes links for 'Hive Tutorials', 'HiPal API', 'HiPal Training', 'HiPal FAQ', 'Submit a POC to Hive', 'Report problems or ask questions', and 'HiPal Home'.

The main area has tabs for 'Query', 'Table', 'Partitions', 'Data Shards (Preview)', 'Cat/Export Data', and 'Import Data'. A 'Table Description' field is present with a note: 'Description not available. Please add here with description.' Below it are fields for 'Export Users' (Zhenyu, Paul Yang, Arshin Thawani, Neeti Jain) and 'Type' (0: Metadata).

The 'Select Columns' section allows selecting columns from a table named 'fb\_dt\_lkup'. The 'Job Status' section displays two active jobs:

- Job ID: 100765** (Status: Running, Duration: 2010-01-27 00:48:00 UTC)
- Job ID: 100897** (Status: Running, Duration: 2010-01-27 00:49:00 UTC)

Each job entry shows the query code, progress bar, and detailed status information.

Źródło: <https://developer.facebook.com>

# Język zapytań Hive

- Filozofia
  - SQL
  - Map-Reduce z własnymi skryptami (hadoop streaming)
- Operatory zapytań
  - Projekcja (select)
  - Równo-złączenia (select \* from t1, t2 where t1.pole1=t2.pole2)
  - Grupowanie (group by)
  - Próbkowanie (select \* from t1 where t1.row...)
  - Kolejność (order by)

# Hive QL – Niestandardowe Mapy /Skrypty Reduce

- Rozszerzony SQL:
  - FROM (
    - FROM pv\_users
    - **MAP** pv\_users.userid, pv\_users.date
    - **USING** 'map\_script' AS (dt, uid)
    - **CLUSTER BY** dt) map
  - INSERT INTO TABLE pv\_users\_reduced
    - **REDUCE** map.dt, map.uid
    - **USING** 'reduce\_script' AS (date, count);
- Map-Reduce: podobny do strumieniowania hadoop

# Skrypt Hadoop Streaming Mapper

```
SELECT TRANSFORM(page_views.userid,  
                  page_views.date)  
USING 'map_script.py'  
AS dt, uid CLUSTER BY datetime  
FROM page_views;
```



## Programowanie aplikacji w chmurze obliczeniowej

### Chmura komputerowa Amazon

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni



**Fundusze  
Europejskie**  
Wiedza Edukacja Rozwój



**Rzeczpospolita  
Polska**

**Unia Europejska**  
Europejski Fundusz Społeczny

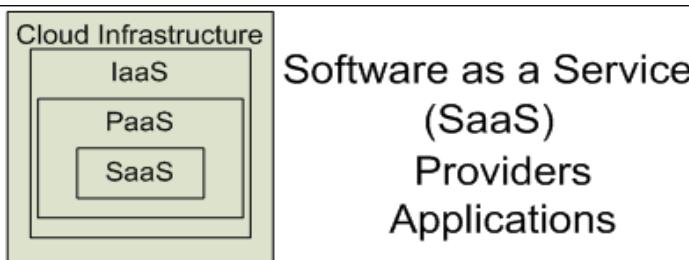
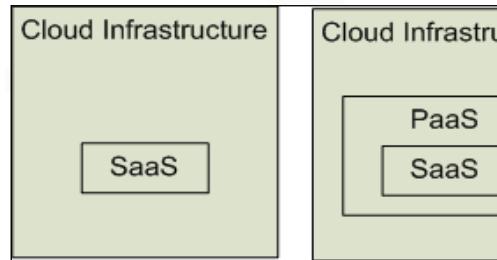


# Modele usług chmurowych

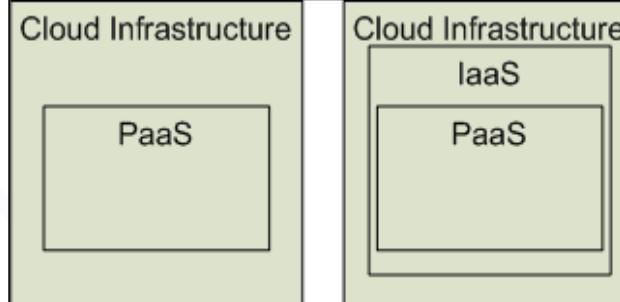
Software as a Service (SaaS)

Platform as a Service (PaaS)

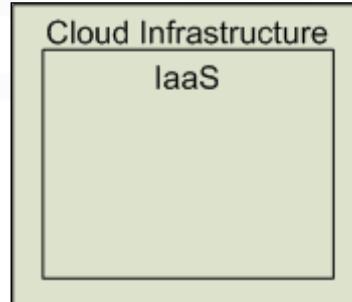
Infrastructure as a Service (IaaS)



Software as a Service (SaaS)  
Providers  
Applications



Platform as a Service (PaaS)  
Deploy customer created Applications



Infrastructure as a Service (IaaS)  
Rent Processing, storage, N/W capacity & computing resources

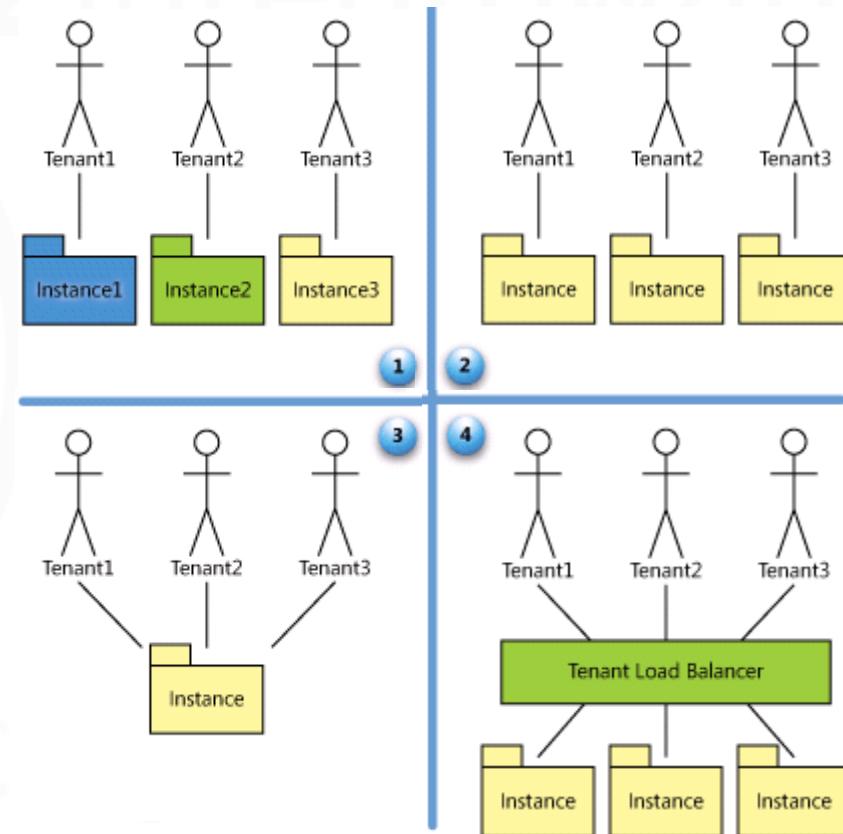
# Model rozwoju SaaS

Poziom 1: Ad-Hoc/Custom – jedna instancja na klienta

Poziom 2: konfigurowalny na klienta

Poziom 3: konfigurowalny i Wielo-Agentowo wydajny

Poziom 4: skalowalny, konfigurowalny i Wielo-Agentowo wydajny

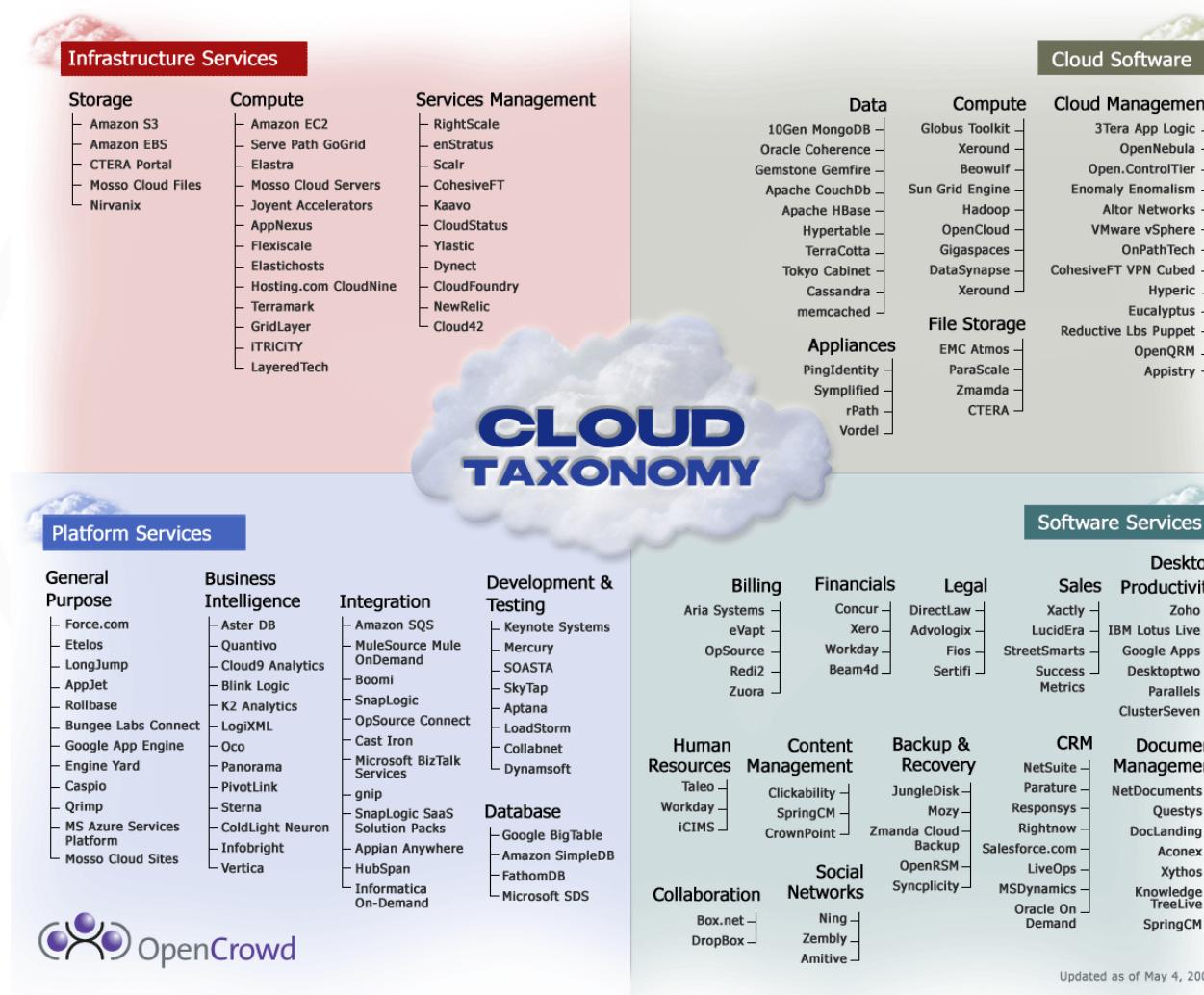


Źródło: Frederick Chong and Gianpaolo Carraro, "Architectures Strategies for Catching the Long Tail"

# Różne warstwy przetwarzania w chmurze

<b>Application Service (SaaS)</b>	MS Live/Exchange Labs, IBM, Google Apps, Salesforce.com Zoho, Cisco
<b>Application Platform</b>	Google App Engine, Force.com, Facebook, Heroku, Amazon Web Services
<b>Server Platform</b>	3Tera, Eucaliptus EC2, GoGrid, RightScale, Linode
<b>Storage Platform</b>	Amazon S3, Dell, Apple, ...

# Taksonomia chmurowa



Źródło: OpenCrowd

# Taksonomia chmurowa (1)

## Infrastructure Services

### Storage

- Amazon S3
- Amazon EBS
- CTERA Portal
- Mosso Cloud Files
- Nirvanix

### Compute

- Amazon EC2
- Serve Path GoGrid
- Elastrata
- Mosso Cloud Servers
- Joyent Accelerators
- AppNexus
- Flexiscale
- Elastichosts
- Hosting.com CloudNine
- Terramark
- GridLayer
- iTRiCiTY
- LayeredTech

### Services Management

- RightScale
- enStratus
- Scalr
- CohesiveFT
- Kaavo
- CloudStatus
- Ylastic
- Dynect
- CloudFoundry
- NewRelic
- Cloud42

Źródło: OpenCrowd

# Taksonomia chmurowa

## Cloud Software

Data	Compute	Cloud Management
10Gen MongoDB	Globus Toolkit	3Tera App Logic
Oracle Coherence	Xeround	OpenNebula
Gemstone Gemfire	Beowulf	Open.ControlTier
Apache CouchDb	Sun Grid Engine	Enomaly Enomalism
Apache HBase	Hadoop	Altair Networks
Hypertable	OpenCloud	VMware vSphere
TerraCotta	Gigaspaces	OnPathTech
Tokyo Cabinet	DataSynapse	CohesiveFT VPN Cubed
Cassandra	Xeround	Hyperic
memcached		Eucalyptus
Appliances	File Storage	Reductive Lbs Puppet
PingIdentity	EMC Atmos	OpenQRM
Symplyfied	ParaScale	Appistry
rPath	Zmamda	
Vordel	CTERA	

Źródło: OpenCrowd

# Taksonomia chmurowa

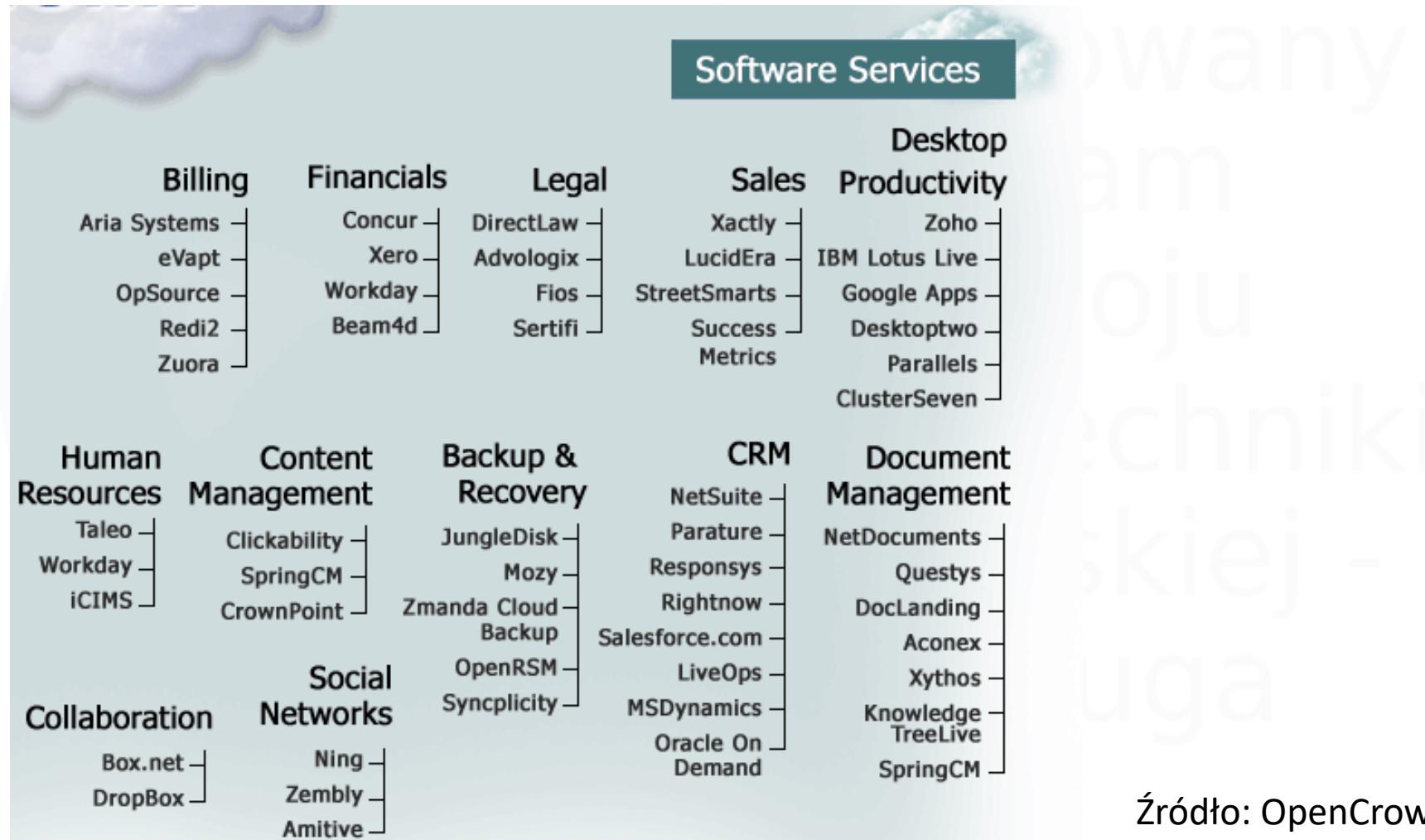
## Platform Services

General Purpose	Business Intelligence	Integration	Development & Testing	Database
Force.com Etelos LongJump AppJet Rollbase Bungee Labs Connect Google App Engine Engine Yard Caspio Qrimp MS Azure Services Platform Mosso Cloud Sites	Aster DB Quantivo Cloud9 Analytics Blink Logic K2 Analytics LogiXML Oco Panorama PivotLink Sterna ColdLight Neuron Infobright Vertica	Amazon SQS MuleSource Mule OnDemand Boomi SnapLogic OpSource Connect Cast Iron Microsoft BizTalk Services gnip SnapLogic SaaS Solution Packs Appian Anywhere HubSpan Informatica On-Demand	Keynote Systems Mercury SOASTA SkyTap Aptana LoadStorm Collabnet Dynamsoft	Google BigTable Amazon SimpleDB FathomDB Microsoft SDS



Źródło: OpenCrowd

# Taksonomia chmurowa



Źródło: OpenCrowd

# Amazon Simple Storage Service (S3)

- Nielimitowane zasoby dyskowe.
- Płać za to co używasz:
  - \$0.20 za GByte przetransferowanych danych,
  - \$0.15 za GByte-Miesiąc za użyte zasoby dyskowe,
  - Update gry Second Life :
    - 1TBytes, 40 000 pobrań w 24 godz. - \$200,



# Amazon EC2

- Amazon Elastic Compute Cloud (EC2):
  - Elastyczne rozwiązanie, zarządzanie 1 do 100+ PCs poprzez WS,
  - Specyfikacja maszyn,
  - Stosunkowo tanie
- Napędzany przez Xen – maszynę wirtualną która:
  - W odróżnieniu od VMware i VPC wykorzystuje "para-wirtualizację", gdzie system-gosć jest modyfikowany, aby używać specjalnych hiper-wywołań:
  - Sprzęt Intel (VT-x/Vanderpool) i AMD (AMD-V).
  - Wspiera “Live Migration” maszyn wirtualnych między hostami.
- Linux, Windows, OpenSolaris
- Konsola zarządzania (aplikacja)

# EC2 - podstawy

- Załadować obraz na S3 i zarejestrować go.
- Uruchomić obraz poprzez Serwis Internetowy.
- Otworzyć wymagane porty dla obrazu.
- Połączyć się z obrazem przez SSH.
- Uruchomić własną aplikację.

# Historia AWS

- Lipiec 2002: uruchomienie Amazon Web Services
  - Witryny osób trzecich mogą wyszukiwać i wyświetlać produkty ze strony internetowej Amazon, dodawać elementy do koszyków Amazon
  - Dostępne przez XML oraz SOAP
- Marzec 2006: Amazon S3 uruchomiony
  - Innowacyjny model rozliczeń „pay-per-use”, który stał się standardem w usługach chmurowych
  - Tańsze niż wiele małych/średnich rozwiązań składowania danych: \$0.15/GB/mies. Składowania danych, \$0.20/GB/mies. za ruch sieciowy
  - Amazon nie jest już czystym detalistą, to miejsce wprowadzania technologii
- Wrzesień 2006: uruchomienie EC2
  - Rdzeniowa infrastruktura obliczeniowa staje się dostępna

# Historia – upowszechnianie usług

- Kwiecień 2008: uruchomienie Google App Engine
  - Niektóre rozwiązania Google używa do swoich celów: Bigtable, GFS, automatyczne skalowanie, itp.
- Listopad 2009: uruchomienie Windows Azure Beta
  - Powszechnie dostępne w 21 krajach w Lutym 2010
  - usługi online Microsoft stopniowo przeniesione do Azure
- Grudzień 2013: uruchomienie Google Compute Engine
  - Zapewnia obsługę niższym poziomie vs. App Engine; pełny zestaw usług
  - Znacznie niższe ceny, szybko dopasowane AWS i Azure

# Zalety Amazon AWS

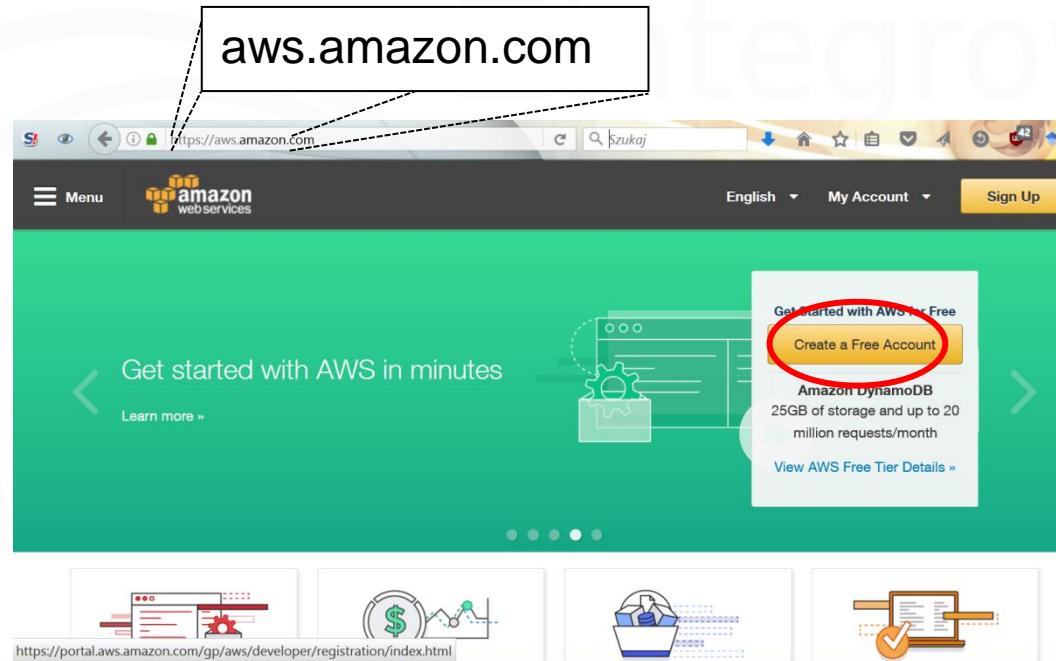
- Amazon jest jednym z wielu dostawców usług chmurowych
  - Inni Microsoft Azure, Google Cloud Engine / App Engine, ...
- Brak (na razie) powszechnego standardu
  - Wstępnie, MS i Google wspierają PaaS (odpowiednio .NET oraz Java)
  - Stopniowo każde rozwiązanie wzrosło do wspierania zarówno IaaS jak i PaaS
  - AWS to PaaS/IaaS z możliwością wyboru wielu ścieżek

# Co to Amazon AWS?

- Amazon Web Services (AWS) zapewniają dostęp do wielu serwisów włączając:
  - Amazon Elastic Compute Cloud (EC2)  
Maszyny wirtualne do uruchamiania własnych rozwiązań programowych
  - Amazon Simple Storage Service (S3)  
Magazyn klucz-wartość, dostępny przez Web
  - Amazon DynamoDB  
Rozproszona baza NoSQL, jedna z wielu w AWS
  - Amazon Elastic MapReduce  
Skalowalne obliczenia MapReduce
  - Amazon Mechanical Turk (MTurk)  
crowdsourcing prostych zadań
  - Amazon SimpleDB  
Prosta baza NoSQL

Używane do projektów

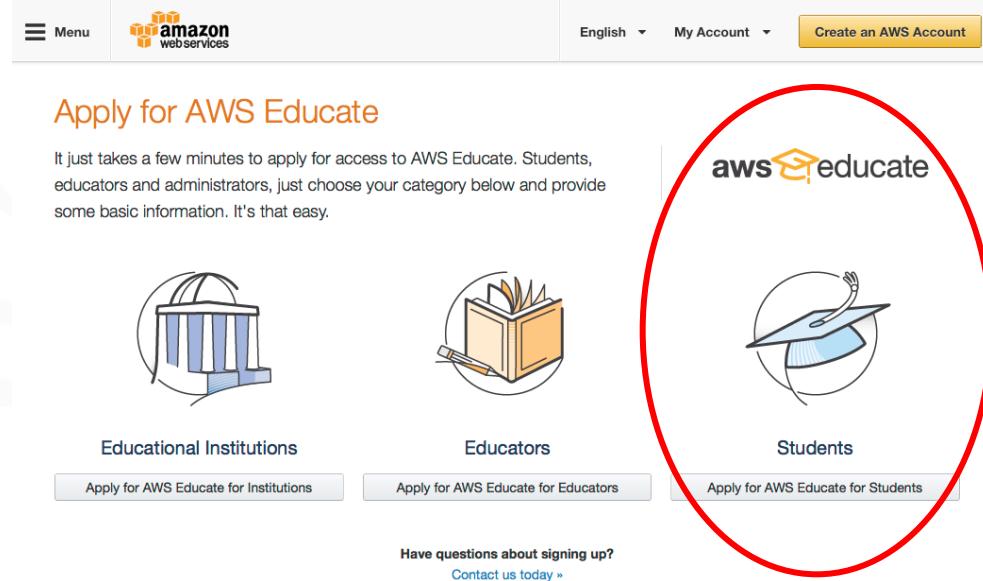
# Założenie konta w AWS



- Aby się zapisać na aws.amazon.com
  - Należy wybrać użytkownika i hasło
  - Służą tylko do interfejsu zarządzania
  - Twoje programy używają innego sposobu rozliczeń (par kluczy RSA, kluczy dostępu, ...) aby współpracować z AWS

# Zapisanie do AWS Educate

- Uzupełnij formularz dostępny na [www.awseducate.com](http://www.awseducate.com)
  - Zakłada że masz już konto w AWS
  - Adres e-mail w domenie uczelni
  - Wypełnienie formularza w.g. Amazon to 2-5 min. (FALSE)
- Ta czynność powinna dać \$100/rok kredytów AWS



# Rozliczenia AWS



Sign In or Create an AWS Account

You may sign in using your existing Amazon.com account or you can create a new account by selecting "I am a new user."

My e-mail address is: \_\_\_\_\_

I am a new user.

I am a returning user  
and my password is:

[Sign in using our secure server](#)

[Forgot your password?](#)

[Has your e-mail address changed?](#)

Learn more about [AWS Identity and Access Management](#) and [AWS Multi-Factor Authentication](#), features that provide additional security for your AWS Account.

## Rozliczenia za logowanie

Witryna AWS i konsola zarządzania

Połączenie z instancją np. przez SSH

## Pary kluczy EC2

- Po co tyle sposobów rozliczeń?

## Access Credentials

There are three types of access credentials used to authenticate your requests to AWS services: (a) access keys, (b) X.509 certificates, and (c) key pairs. Each access credential type is explained below.

[Access Keys](#) [X.509 Certificates](#) [Key Pairs](#)

Use X.509 certificates to make secure SOAP protocol requests to AWS service APIs.

Exceptions: Amazon S3 and Amazon Mechanical Turk instead require your [Access Keys](#) for SOAP requests.

Created	X.509 Certificate	Status
[REDACTED]	cert-[REDACTED].pem	Active (Make Inactive)

[Create a new Certificate](#) | [Upload Your Own Certificate](#)

For your protection, AWS doesn't ask for your private key or retain it on file. You should also never share your private key with anyone. In addition, industry best practice recommends frequent certificate rotation.

[Learn more about X.509 Certificates](#)

## Narzędzia wiersza polecenia, API SOAP

## Certyfikaty X.509

## Access Credentials

There are three types of access credentials used to authenticate your requests to AWS services: (a) access keys, (b) X.509 certificates, and (c) key pairs. Each access credential type is explained below.

[Access Keys](#) [X.509 Certificates](#) [Key Pairs](#)

Use access keys to make secure REST or Query protocol requests to any AWS service API. We create one for you when your account is created — see your access key below.

Created	Access Key ID	Secret Access Key	Status
[REDACTED]	[REDACTED]	Show	Active (Make Inactive)

[Create a new Access Key](#)

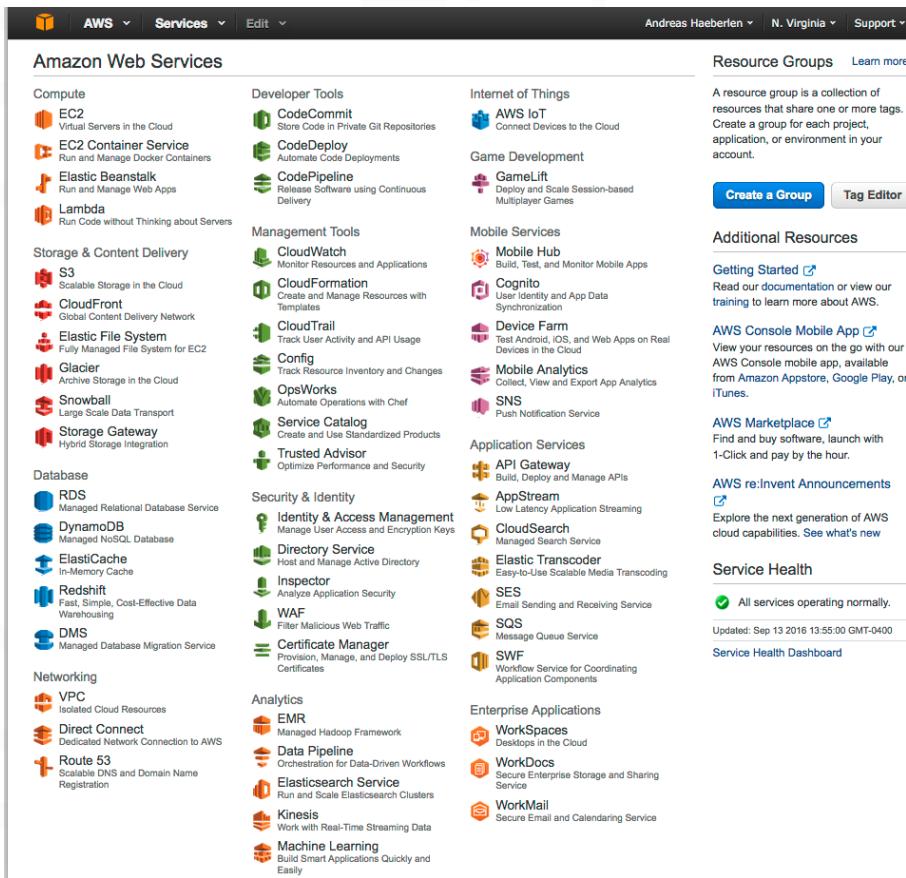
For your protection, you should never share your secret access keys with anyone. In addition, industry best practice recommends frequent key rotation.

[Learn more about Access Keys](#)

## API REST

## Klucze dostępowe

# Konsola zarządzania AWS



- Służy do kontroli wielu serwisów AWS:
  - Na przykład, start/stop instancji EC2, tworzenie kontenerów S3

# REST i SOAP

- Uzyskanie dostępu do AWS przez program:
  - Przykład: Uruchomićinstancję EC2, zapisać wartości w S3
  - Komunikacją -> protokoły REST
- Representational State Transfer (REST)
  - Zapytanie wysłane przez HTTP (GET lub POST); każdy parametr jest kodowany w URL lub zawarty w ciele
  - Odpowiedź w postaci struktury danych XML dostarczonej przez HTTP
- Wcześniej: Simple Object Access Protocol (SOAP)
  - Nie takie proste jak sugeruje nazwa
  - Bazuje na XML, rozszerzalny, ogólny, znormalizowany, ale również nieco nadmiarowy
  - Wsparcie do grudnia 2014

# REST

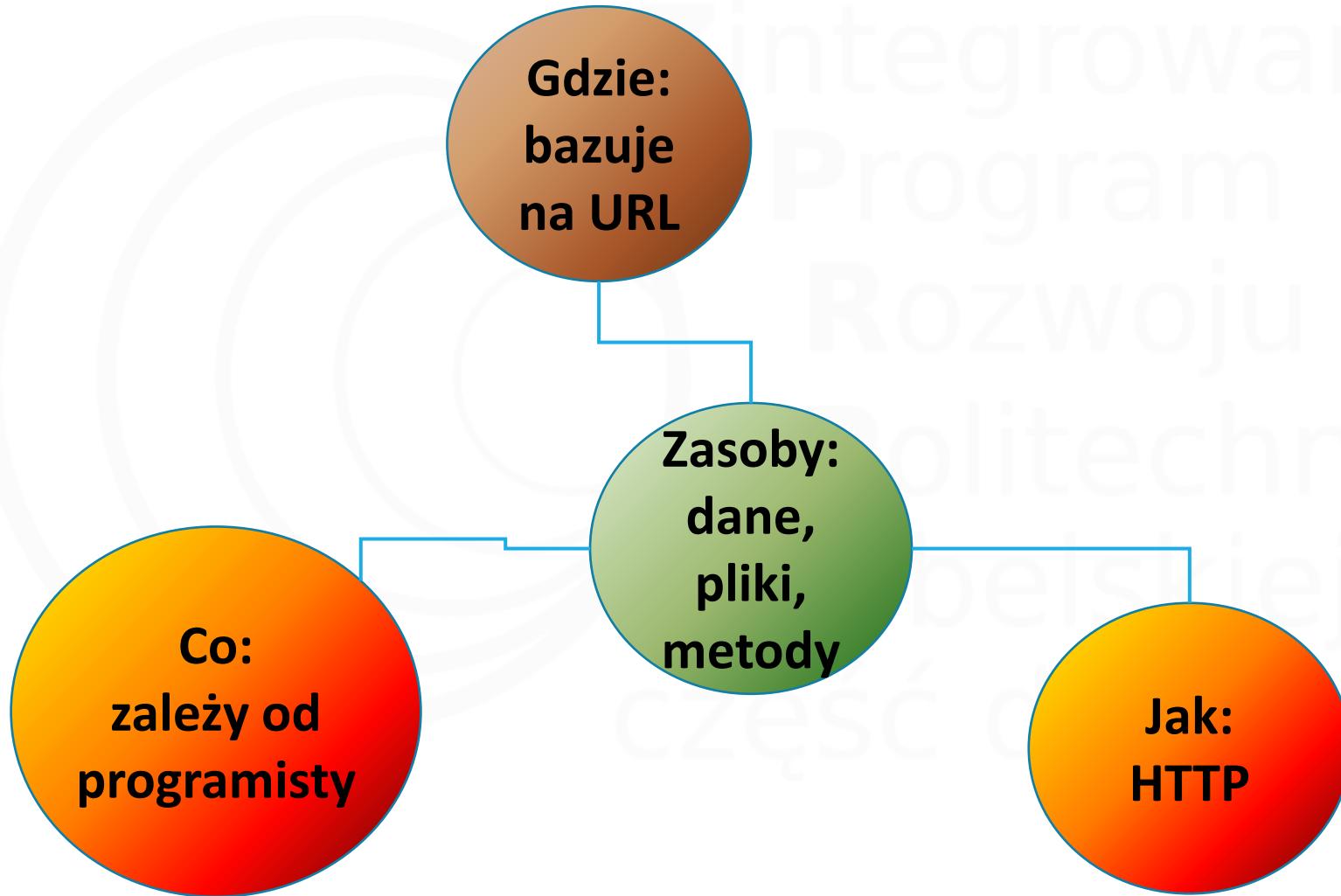
- **Co to jest REST?**

- Rodzaj architektury programowej dla systemów rozproszonych

- **Kto/Gdzie/Kiedy?**

- Po raz pierwszy wprowadzony w doktoracie Roya Fieldinga w 2000 roku
- Roy Fielding:  
Jeden z głównych autorów dokumentacji HTTP

# REST – rdzeń główny



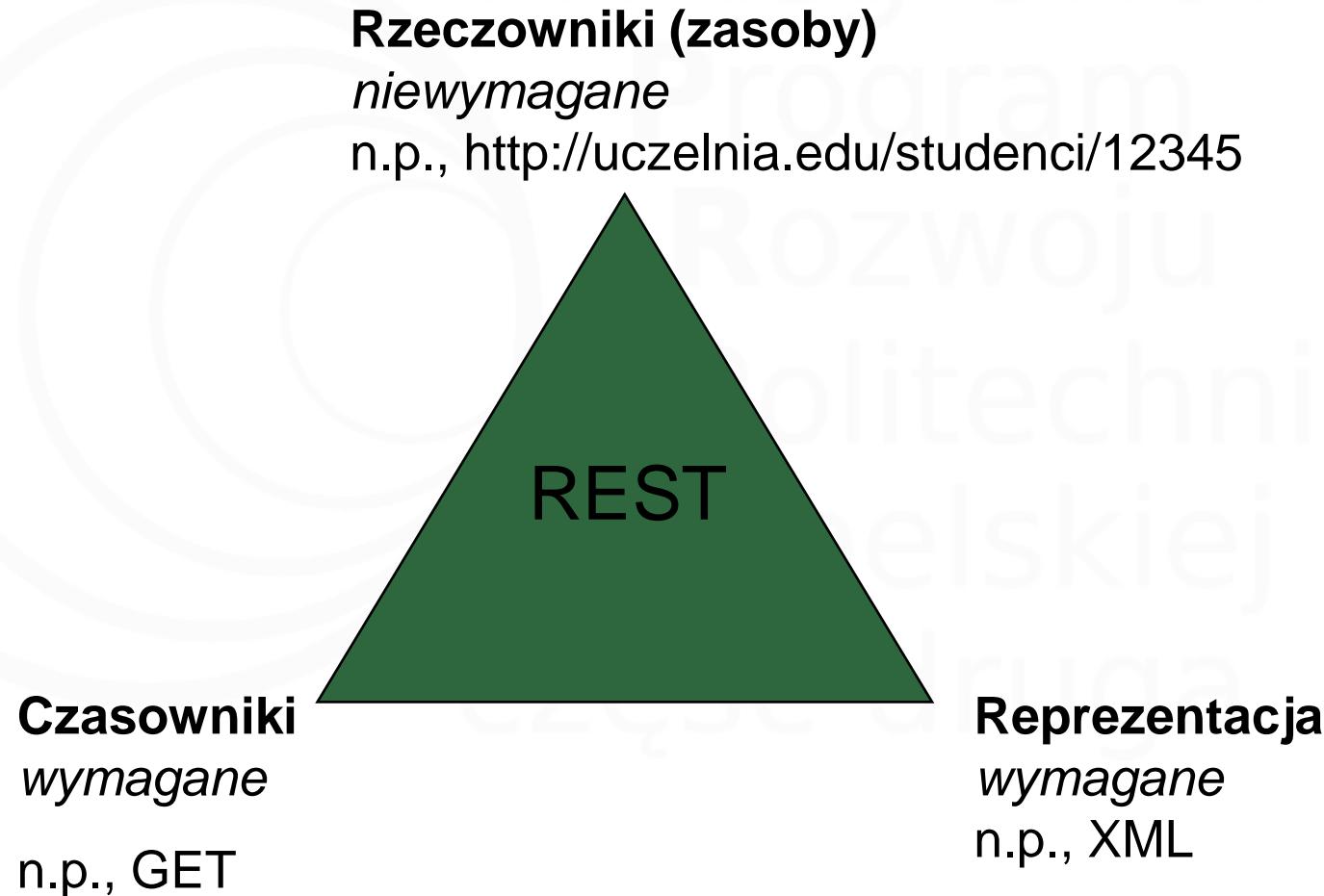
# REST i HTTP

- Motywacją dla powstania REST było uchwycić cechy sieci Web, które przyniosły sukces.
  - zasoby adresowane przez URI
  - używa protokołu HTTP
  - Utwórz Zapytanie -> Otrzymaj odpowiedź  
->Wyświetl Odpowiedź
- Wykorzystuje użycie protokołu HTTP poza HTTP POST i HTTP GET
  - HTTP PUT, HTTP DELETE

# REST – to nie standard

- REST nie jest standarde,
  - JSR 311: JAX-RS: The JavaTM API for RESTful Web Services
- Ale używa wielu standardów:
  - HTTP
  - URL
  - XML/HTML/GIF/JPEG/itp. (reprezentacja zasobów)
  - text/xml, text/html, image/gif, image/jpeg, itd. (typy zasobów, typy MIME)

# Główne idee



# Zasoby

- Kluczową abstrakcją informacji w REST jest zasób.
- Zasób jest koncepcyjnym mapowanie do zbioru podmiotów
  - Wszelkie informacje, które mogą być nazwane mogą być zasobem: dokument lub obraz, usługa czasowa (n.p. dzisiejsza pogoda ), zbiór innych zasobów, obiekt nie-wirtualny (na przykład osoba)...
- Zasób jest reprezentowany za pomocą globalnego identyfikatora (URI w HTTP)
  - <https://cloud.google.com/storage/docs/xml-api/reference-uris>

# Nazewnictwo zasobów

- REST używa URI do identyfikacji zasobów
  - <http://localhost/ksiazki/>
  - <http://localhost/ksiazki/ISBN- 1234>
  - <http://localhost/ksiazki/ISBN-1234/autorzy>
  - <http://localhost/zajecia>
  - <http://localhost/zajecia/pwcho>
  - <http://localhost/zajecia/pwcho/studenci>
- Podczas przemierzania ścieżki z bardziej ogólnej do bardziej szczegółowej, nawigujemy do danych.

# Czasowniki

- Reprezentują akcję przeprowadzaną na zasobach
- HTTP GET
- HTTP POST
- HTTP PUT
- HTTP DELETE

# HTTP GET

- Jak klienci pytają o informacje, których poszukuję.
- Wysyłanie żądania GET przesyła dane z serwera do klienta w jakiejś reprezentacji
- GET <http://localhost/ksiazki>
  - Pobierz wszystkie książki
- GET <localhost/ksiazki/ISBN-1234>
  - Pobierz książki z numerem ISBN-1234
- GET <http://localhost/ksiazki/ISBN-1234/autorzy>
  - Pobierz autorów dla książki z numerem ISBN-1234

# HTTP PUT, HTTP POST

- HTTP POST tworzy zasób
- HTTP PUT aktualizuje zasób
- POST <http://localhost/ksiazki/>
  - Zawartość: {tytuł, autorzy[], ...}
  - Tworzy książkę z zadanymi własnościami
- PUT <http://localhost/ksiazki/isbn-1234>
  - Zawartość: {isbn, tytuł, autorzy[], ...}
  - Aktualizuje książkę z numerem ISBN-1234 podanymi wartościami

# HTTP DELETE

- Usuwa zasób podany w URI
- DELETE <http://localhost/ksiazki/ISBN-0123>
  - Usuwa książkę o numerze ISBN-0123

# Reprezentacje danych

- Sposób reprezentacji danych lub zwracane do klienta do prezentacji.
- Dwa główne formaty:
  - JavaScript Object Notation (JSON)
  - XML
- Powszechnie jest mieć wiele reprezentacji tych samych danych.

# Reprezentacje



- **XML**

- <ZAJĘCIA>
  - <ID>PWCHO</ID>
  - <NAME>Amazon Web Services</NAME>
- </ZAJĘCIA>

- **JSON**

- {zajęcia
  - {id: PWCHO}
  - {name: Amazon Web Services}
- }

# Zapytanie HTTP/Odpowiedź jako REST



# Przykład REST w AWS

Parametry

https://sdb.amazonaws.com/?Action=PutAttributes  
&DomainName=MyDomain  
&ItemName=Item123  
&Attribute.1.Name=Color&Attribute.1.Value=Blue  
&Attribute.2.Name=Size&Attribute.2.Value=Med  
&Attribute.3.Name=Price&Attribute.3.Value=0014.99  
&AWSAccessKeyId=<valid\_access\_key>  
&Version=2009-04-15  
&Signature=[valid signature]  
&SignatureVersion=2  
&SignatureMethod=HmacSHA256  
&Timestamp=2010-01-25T15%3A01%3A28-07%3A00

Przykładowe zapytanie

Wywołana metoda

Elementy odpowiedzi

<PutAttributesResponse>  
<ResponseMetadata>  
<StatusCode>Success</StatusCode>  
<RequestId>f6820318-9658-4a9d-89f8  
b067c90904fc</RequestId>  
<BoxUsage>0.0000219907</BoxUsage>  
</ResponseMetadata>  
</PutAttributesResponse>

Przykładowa odpowiedź

Źródło: <http://awsdocs.s3.amazonaws.com/SDB/latest/sdb-dg.pdf>

# Dla porównania SOAP

```
<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV='http://schemas.xmlsoap.org/soap/envelope/'
    xmlns:SOAP-ENC='http://schemas.xmlsoap.org/soap/encoding/'
    xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
    xmlns:xsd='http://www.w3.org/2001/XMLSchema'>
    <SOAP-ENV:Body>
        <PutAttributesRequest xmlns='http://sdb.amazonaws.com/doc/2009-04-15'>
            <Attribute><Name>a1</Name><Value>2</Value></Attribute>
            <Attribute><Name>a2</Name><Value>4</Value></Attribute>
            <DomainName>domain1</DomainName>
            <ItemName>eID001</ItemName>
            <Version>2009-04-15</Version>
        </PutAttributesRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Przykładowe zapytanie

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Body>
        <PutAttributesResponse>
            <ResponseMetadata>
                <RequestId>4c68e051-fe45-43b2-992a-a24017ffe7ab</RequestId>
                <BoxUsage>0.0000219907</BoxUsage>
            </ResponseMetadata>
        </PutAttributesResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Przykładowa odpowiedź

Źródło: <http://awsdocs.s3.amazonaws.com/SDB/latest/sdb-dg.pdf>

# Amazon EC2

- Infrastructure-as-a-Service (IaaS)
  - Można wypożyczać różnego rodzaju maszyny na godziny
  - We własnych instancjach maszyn VM (Linux/Windows) można uruchamiać programy
    - Np: serwer Web, silnik wyszukiwawczy, rendering filmu, itp.

0.5 GB pamięć  
1 wirtualny rdzeń  
(ECU zmienne)  
P.masowa: EBS

	ECU	Memory (GiB)	Instance Storage (GB)	Linux/UNIX Usage
<b>General Purpose - Current Generation</b>				
t2.micro	1	Variable	1	EBS Only \$0.013 per Hour
t2.small	1	Variable	2	EBS Only \$0.026 per Hour
t2.medium	2	Variable	4	EBS Only \$0.052 per Hour
t2.large	2	Variable	8	EBS Only \$0.104 per Hour
m4.large	2	6.5	8	EBS Only \$0.126 per Hour
m4.xlarge	4	13	16	EBS Only \$0.252 per Hour
m4.2xlarge	8	26	32	EBS Only \$0.504 per Hour
m4.4xlarge	16	53.5	64	EBS Only \$1.008 per Hour
m4.10xlarge	40	124.5	160	EBS Only \$2.52 per Hour
m3.medium	1	3	3.75	1 x 4 SSD \$0.067 per Hour
m3.large	2	6.5	7.5	1 x 32 SSD \$0.133 per Hour
m3.xlarge	4	13	15	2 x 40 SSD \$0.266 per Hour
m3.2xlarge	8	26	30	2 x 80 SSD \$0.532 per Hour
<b>Compute Optimized - Current Generation</b>				
c4.large	2	8	3.75	EBS Only \$0.11 per Hour
c4.xlarge	4	16	7.5	EBS Only \$0.22 per Hour
c4.2xlarge	8	31	15	EBS Only \$0.441 per Hour
c4.4xlarge	16	62	30	EBS Only \$0.882 per Hour
c4.8xlarge	36	132	60	EBS Only \$1.763 per Hour
c3.large	2	7	3.75	2 x 16 SSD \$0.105 per Hour
c3.xlarge	4	14	7.5	2 x 40 SSD \$0.21 per Hour
c3.2xlarge	8	28	15	2 x 80 SSD \$0.42 per Hour
c3.4xlarge	16	55	30	2 x 160 SSD \$0.84 per Hour
c3.8xlarge	32	108	60	2 x 320 SSD \$1.68 per Hour
<b>GPU Instances - Current Generation</b>				
g2.2xlarge	8	26	15	60 SSD \$0.65 per Hour
g2.8xlarge	32	104	60	2 x 120 SSD \$2.6 per Hour
<b>Memory Optimized - Current Generation</b>				
r3.large	2	6.5	15	1 x 32 SSD \$0.175 per Hour
r3.xlarge	4	13	30.5	1 x 80 SSD \$0.35 per Hour
r3.2xlarge	8	26	61	1 x 160 SSD \$0.7 per Hour
r3.4xlarge	16	52	122	1 x 320 SSD \$1.4 per Hour
r3.8xlarge	32	104	244	2 x 320 SSD \$2.8 per Hour
<b>Storage Optimized - Current Generation</b>				
d2.xlarge	4	14	30.5	1 x 800 SSD \$0.853 per Hour
d2.2xlarge	8	27	61	2 x 800 SSD \$1.705 per Hour
d2.4xlarge	16	53	122	4 x 800 SSD \$3.41 per Hour
d2.8xlarge	32	104	244	8 x 800 SSD \$6.82 per Hour
d2.xlarge	4	14	30.5	3 x 2000 HDD \$0.69 per Hour
d2.2xlarge	8	28	61	6 x 2000 HDD \$1.38 per Hour
d2.4xlarge	16	56	122	12 x 2000 HDD \$2.76 per Hour
d2.8xlarge	36	116	244	24 x 2000 HDD \$5.52 per Hour

Źródło: <http://aws.amazon.com/ec2/#pricing> (9/10//2015)

# Bezpieczeństwo

- Klienci amerykańskiej firmy jubilerskiej Limoges Jewelry nie kryli zdziwienia, kiedy dowiedzieli się, że ich wrażliwe dane są dostępne w przestrzeni publicznej. Prawdopodobnie słabe zabezpieczenia serwisu Amazon Web Services S3 sprawiły, że przestępcy otrzymali na tacy 1,3 miliona rekordów zawierających nazwiska, adresy, kody pocztowe, numery telefonów, a także hasła tekstowe.
- Według badaczy bezpieczeństwa z Kromtech, którzy wykryli tę anomalię, baza danych należy do MBM Company - firmy jubilerskiej, która sprzedaje swoje produkty pod marką Limoges Jewelry. Ich zdaniem informacje mogły być dostępne publicznie od 13 stycznia 2018 roku. Jeśli eksperci do bezpieczeństwa wpadli na ten trop, istnieje więcej niż duże prawdopodobieństwo, że dotarli do nich również cyberprzestępcy. Na liście partnerów feralnej firmy znajdują się Walmart, Amazon, Sears, Kmart, Target i wiele sklepów internetowych. Afera związana z wyciekiem danych negatywnie wpłynęła na reputację Walmartu.

Źródło: *Bitdefender 23.03.2018*



Programowanie aplikacji w chmurze obliczeniowej

Rozwiązania chmurowe firm Google oraz Microsoft

dr hab. inż. Dariusz Czerwiński,  
profesor uczelni





# Google App Engine

- **Google App Engine (GAE)** to usługa Platform as a Service (PaaS) do opracowywania i hostowania aplikacji web przechowywanych w centrach danych zarządzanych przez Google.
- Google App Engine pozwala uruchamiać aplikacja web na infrastrukturze Google.
  - Łatwe w budowie
  - Łatwe w utrzymaniu
  - Łatwa skalowalność przy wzroście zapotrzebowania na ruch i zasoby dyskowe.
- Darmowe do 1 GB zasobów dyskowych i wystarczająca liczba CPU i pasma aby obsłużyć 5 milionów wizyt na witrynie w ciągu miesiąca. 10 aplikacji na konto Google.

# GAE – wsparcie języków programowania

- Java:
- App Engine uruchamia aplikacje JAVA w maszynie wirtualnej JAVA 7 (obecnie JAVA 6 również).
- Używa standardu JAVA Servlet dla aplikacji web:
  - WAR (Web Applications ARchive) struktura katalogów
  - klasy Servlet
  - Java Server Pages (JSP)
  - pliki statyczne i zdanymi
  - deskryptor wdrożenia (web.xml)
  - inne pliki konfiguracyjne
- Odnośnik:  
[https://developers.google.com/appengine/docs/java/getting\\_started/](https://developers.google.com/appengine/docs/java/getting_started/)

# GAE – wsparcie języków programowania

## Python:

- Używa standardu WSGI (Web Server Gateway Interface)
- Aplikacje Python mogą być pisane z użyciem:
  - szkieletu Webapp2
  - szkieletu Django
  - dowolnego kodu python CGI (Common Gateway Interface) standard.
- Można się zapoznać:  
<https://developers.google.com/appengine/docs/python/gettingstartedpython27/>

# GAE – wsparcie języków programowania

- **PHP (Wsparcie eksperymentalne):**

- Lokalne serwery rozwojowe są dostępne dla każdego developera i do testowania.
- Tylko aplikacje znajdujące się na białej mogą być dystrybuowane w GAE.
- (<https://gaeforphp.appspot.com/>).
- Pomoc:  
<https://developers.google.com/appengine/docs/php/>

- **Google's Go:**

- Go to środowisko programistyczne open source programming firmy Google.
- Ściśle związane z Google App Engine.
- Aplikacje mogą być pisane z użyciem App Engine's Go SDK.
- Pomoc:  
<https://developers.google.com/appengine/docs/go/overview>

# GAE – przechowywanie danych

- App Engine Datastore:
  - Obiektowa struktura składowania danych bez schematu NoSQL, z silnikiem zapytań i atomowymi transakcjami.
  - Obiekt danych jest nazywany encją “Entity” i w pewnym sensie podobny do nazwy tabeli (~table name) oraz zestaw atrybutów (~column names).
  - JAVA JDO/ JPA interfejsy oraz interfejsy składowania Python.
- Google cloud dostarcza relacyjny język SQL:
  - Dostarcza relacyjny języka SQL dla składowanych danych.
  - Podobny do MySQL RDBMS.

# Google Cloud Store

- Składowanie danych
- Serwis RESTful dla składowania i pozyskiwania danych.
- Szybkie, skalowalne i łatwo dostępne rozwiązanie.
- Dostarcza wielowarstwowej redundancji. Dane są replikowane w wielu centrach danych naraz.
- Zapewnia wiele różnych poziomów kontroli dostępu.
- API bazujące na HTTP.

# Serwisy GAE

- App Engine zapewnia również szereg usług do wykonywania typowych czynności podczas zarządzania aplikacją.
- **URL Fetch:**
  - Ułatwia dostęp aplikacji do zasobów w Internecie, takich jak usługi sieciowe lub danych.
- **Mail (Poczta):**
  - Ułatwia aplikacji wysyłanie wiadomości e-mail za pomocą infrastruktury Google.
- **Memcache:**
  - Wysoka wydajność magazynowania klucz-wartość w pamięci.
  - Może być używany do przechowywania danych tymczasowych, które nie musi być utrwalone.

# Bezpieczeństwo w GAE - Piaskownica

- **Sandbox:**

- Wszystkie hostowane aplikacje są uruchamiane w bezpiecznym środowisku, które zapewnia ograniczony dostęp do bazowego systemu operacyjnego.
- Piaskownica izoluje aplikację w swoim bezpiecznym, niezawodnym środowisku, które jest niezależne od sprzętu, systemu operacyjnego i fizycznej lokalizacji serwera WWW.

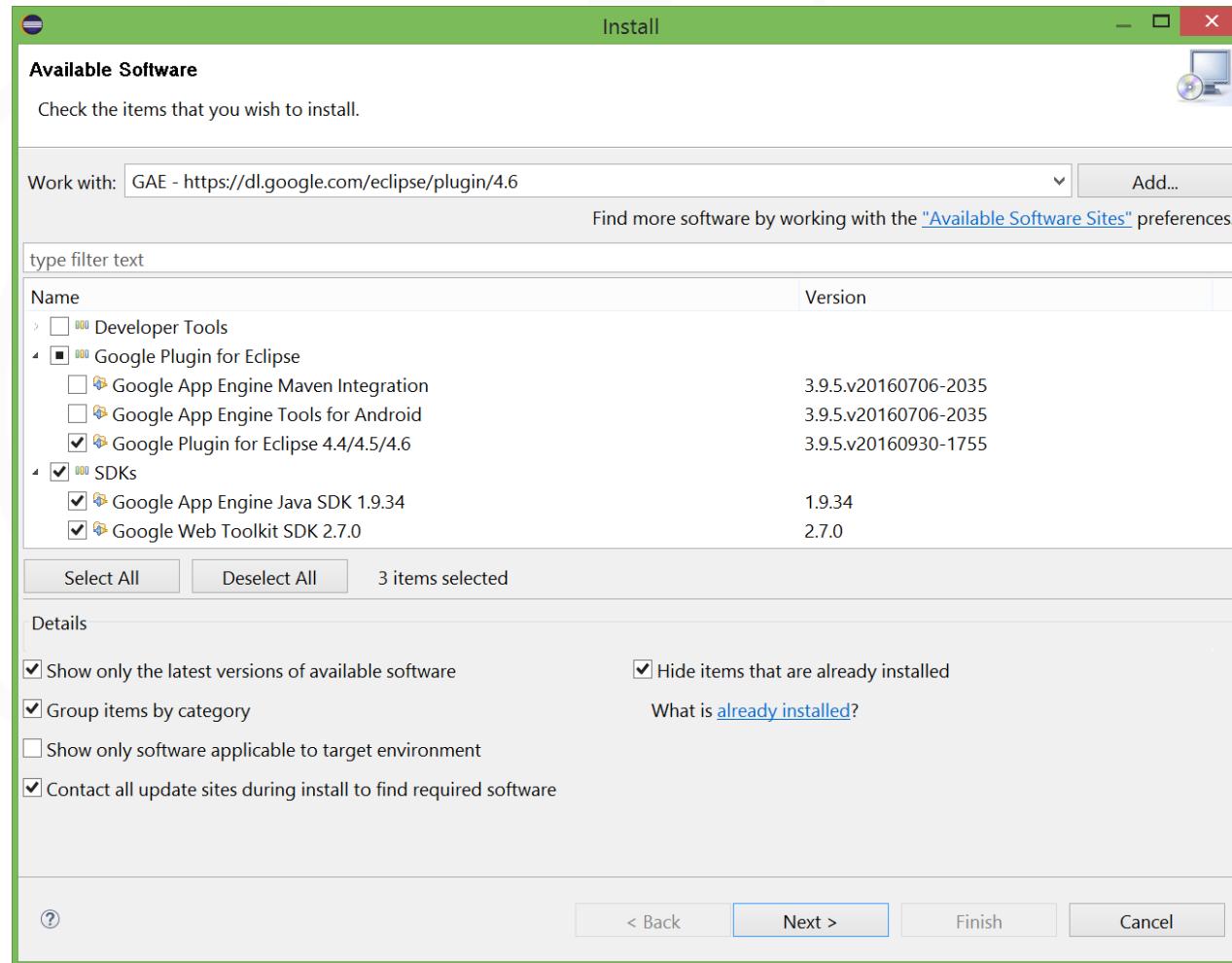
# Bezpieczeństwo w GAE - Piaskownica

- Ograniczenia nałożone przez piaskownicę (dla bezpieczeństwa):
  - Aplikacja może uzyskać dostęp do innych komputerów przez Internet za pomocą dostarczonego URL pobierania i usług e-mail. Inne komputery mogą łączyć się tylko do aplikacji za pośrednictwem protokołu HTTP / HTTPS i zapytań do standardowych portów (80/443).
  - Aplikacje nie mogą zapisywać do lokalnego systemu plików w dowolnym środowisku uruchomieniowym.
  - Kod aplikacji działa tylko w odpowiedzi na żądanie www, w kolejce zadań lub zaplanowanego zadania i musi zwrócić dane odpowiedzi w ciągu 60 sekund. Procedura obsługi zapytania nie może zapoczątkować podprocesu lub wykonania kodu gdy odpowiedź została wysłana.

# Przygotowanie IDE do GAE

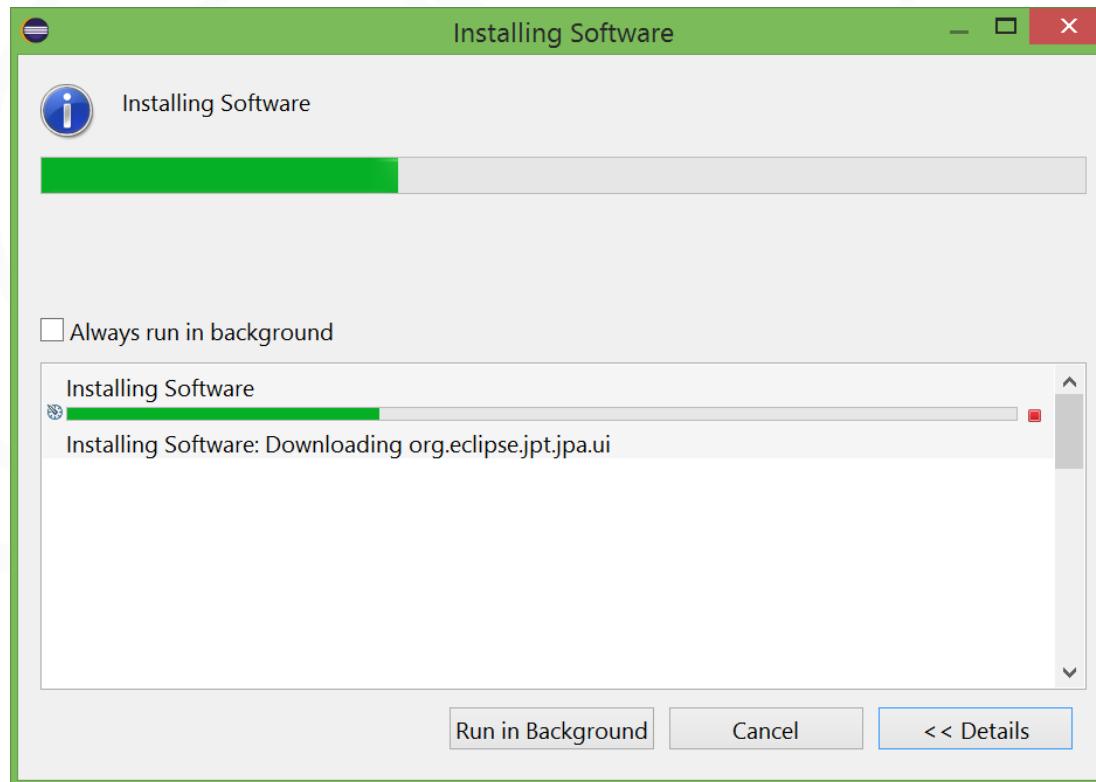
- **Eclipse** – dlaczego?
  - Ponieważ istnieje wtyczka GAE do Eclipse
- **Help -> Install New Software...**
- W oknie dialogowym należy wpisać:
  - <https://dl.google.com/eclipse/plugin/4.6>

# Eclipse plugin

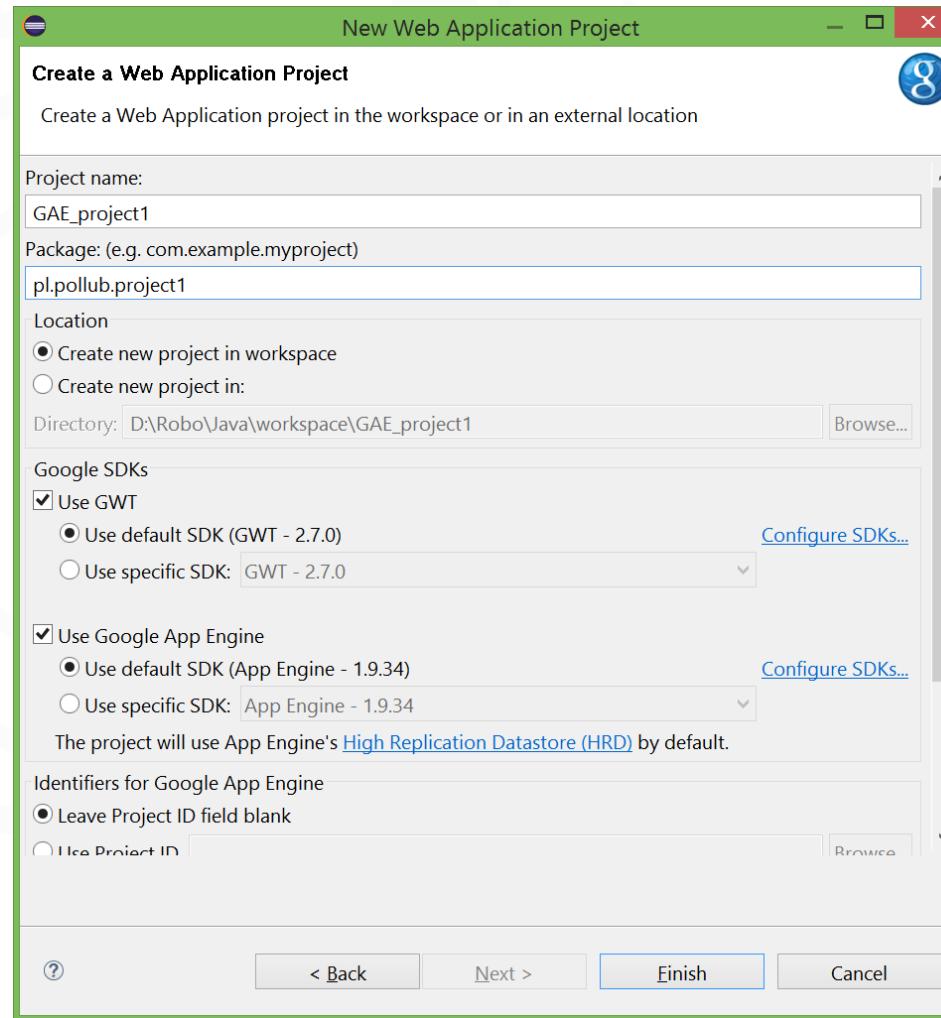


# Eclipse plugin

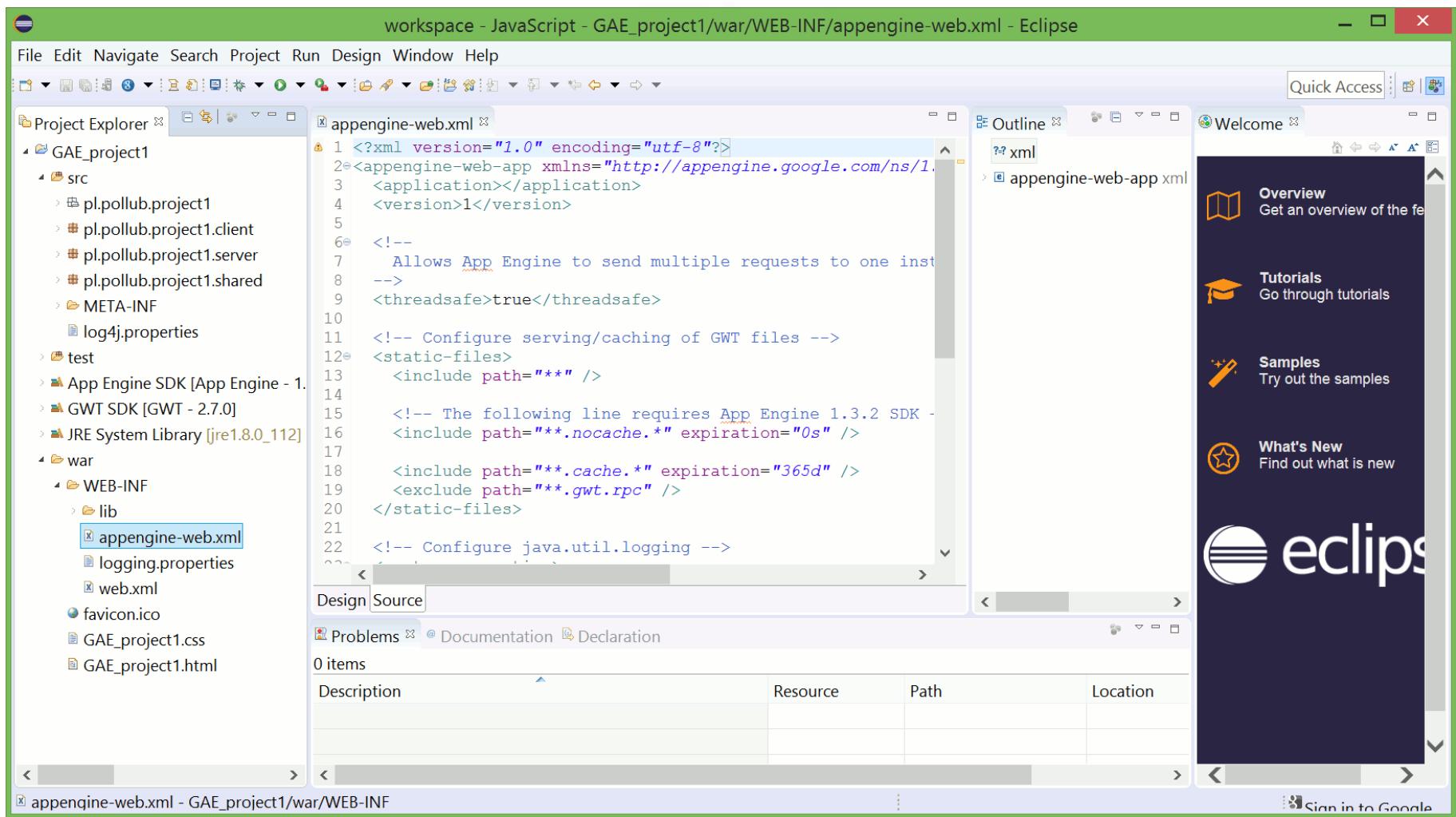
- Gdy pojawi się komunikat bezpieczeństwa ostrzegający przed niepodpisanym certyfikatem należy zatwierdzić OK



# GAE – nowy projekt



# GAE – elementy projektu



# GAE – katalogi w aplikacji

```
Project1/  
    src/  
        ...Kod źródłowy Java...  
        META-INF/  
            ...inne pliki konfiguracyjne...  
    war/  
        ...JSPs, obrazy, pliki danych...  
        WEB-INF/  
            ...konfiguracja aplikacji...  
        lib/  
            ...JARs dla bibliotek...  
    classes/  
        ...skompilowane klasy...
```

# Plik konfiguracyjny GAE

```
*appengine-web.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
3   <application></application>
4   <version>1</version>
5   <!--
6     Allows App Engine to send multiple requests to one instance in parallel:
7   -->
8   <threadsafe>true</threadsafe>
9   <!-- Configure serving/caching of GWT files -->
10  <static-files>
11    <include path="**" />
12
13  <!-- The following line requires App Engine 1.3.2 SDK -->
14  <include path="*.nocache.*" expiration="0s" />
15
16  <include path="*.cache.*" expiration="365d" />
17  <exclude path="*.gwt.rpc" />
18 </static-files>
19 <!-- Configure java.util.logging -->
20 <system-properties>
21   <property name="java.util.logging.config.file" value="WEB-INF/logging.properties"/>
22 </system-properties>
23 <!--
24   HTTP Sessions are disabled by default. To enable HTTP sessions specify:
25   <sessions-enabled>true</sessions-enabled>
26   It's possible to reduce request latency by configuring your application to
27   asynchronously write HTTP session data to the datastore:
28   <async-session-persistence enabled="true" />
29   With this feature enabled, there is a very small chance your app will see
30   stale session data. For details, see
31   https://cloud.google.com/appengine/docs/java/config/appconfig#Java_appengine_web_xml_Enabling_sessions
32 -->
33 </appengine-web-app>
```

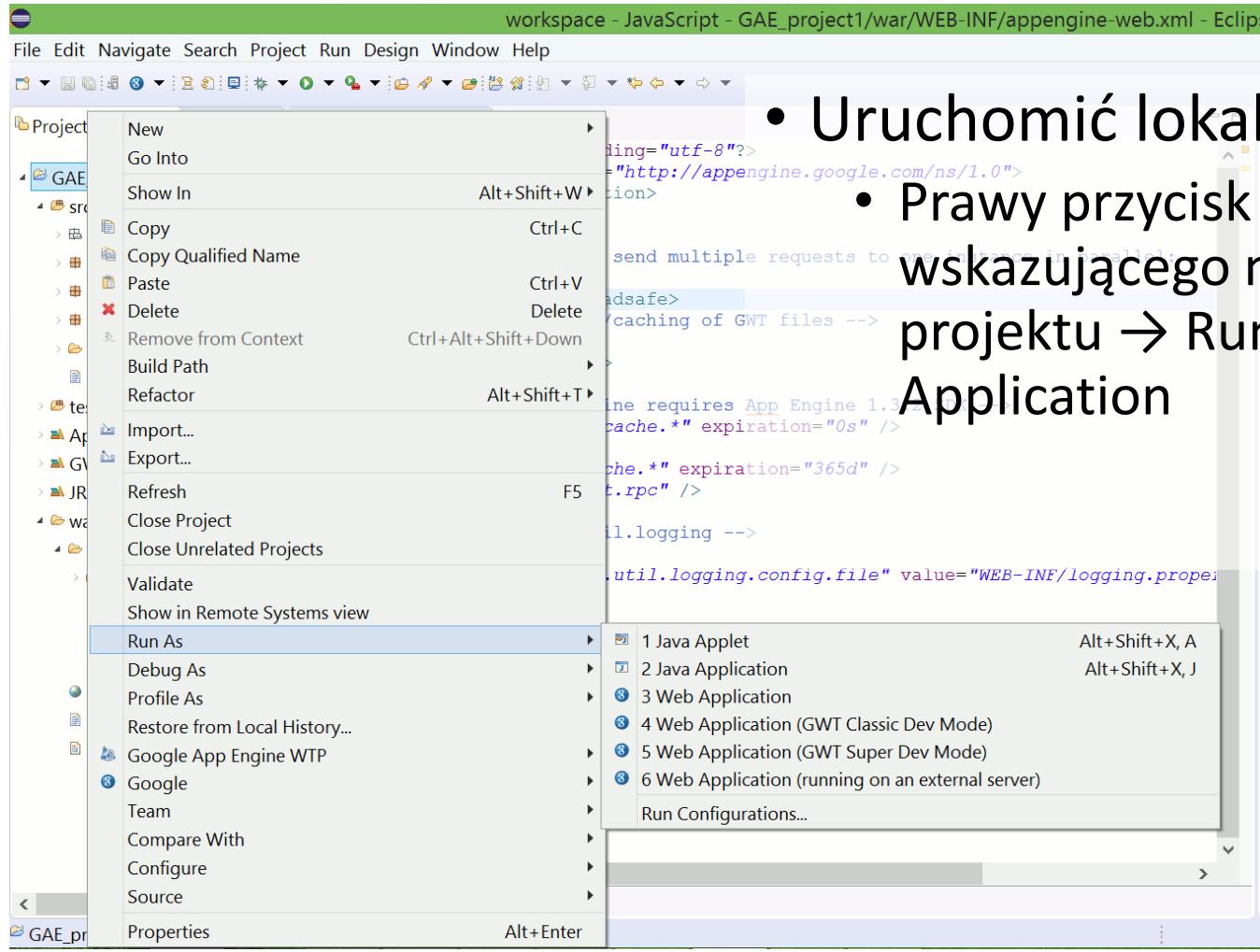
# Plik konfiguracyjny appengine-web.xml

```
<?xml version="1.0" encoding="utf-8"?>
<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
  <application></application>
  <version>1</version>

  <!-- Configure java.util.logging -->
  <system-properties>
    <property name="java.util.logging.config.file"
              value="WEB-
INF/logging.properties"/>
  </system-properties>

</appengine-web-app>
```

# Jak sprawdzić czy działa



- Uruchomić lokalnie:

- Prawy przycisk urządzenia wskazującego na nazwie projektu → Run as → Web Application

# Uruchomienie w GAE

- Założyć konto na <https://appengine.google.com/> i utworzyć ID aplikacji dla aplikacji web.
- W przykładzie podano nazwę projektu „Project123”, pobrano ID z witryny GAE i wstawiono to do ***appengine-web.xml***.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
3   <application>project123-155706</application>
4     <version>1</version>
5
6   <!-- Configure java.util.logging -->
7   <system-properties>
8     <property name="java.util.logging.config.file" value="WEB-INF/logging.properties">
9   </system-properties>
10
11 </appengine-web-app>
```

# Założenie konta w GAE

- Jeżeli użytkownik posiada konto Gmail to wystarczy się zalogować i utworzyć nowy projekt.

The screenshot shows the Google Cloud Platform interface. At the top, there is a banner with the text: "Zarejestruj się, aby skorzystać z bezpłatnej wersji próbnej, a otrzymasz 300 USD w postaci kredytu i 60 dni na poznanie oraz przetestowanie Google Cloud Platform. [Więcej informacji](#)". Below the banner, the main navigation bar includes "Google Cloud Platform" and "Project". The left sidebar is titled "App Engine" and lists various services: Uslugi, Wersje, Instancje, Kolejki zadań, Skany bezpieczeństwa, Limity, Blobstore, Memcache, Wyszukaj, and Ustawienia. The main content area is titled "Panel informacyjny" and contains the message: "App Engine Panel informacyjny" and "Google Cloud Platform używa projektów do zarządzania zasobami. Aby rozpocząć, utwórz projekt." with a blue "Utwórz projekt" button.

# Założenie projektu

The screenshot shows the 'Nowy projekt' (New project) dialog box on the Google Cloud Platform. The dialog box is centered over a blurred background of the GCP dashboard. The dialog contains the following fields:

- Nazwa projektu:** Project1
- Identyfikator projektu:** project1-155706 (with a 'Edytuj' link)
- Opis:** Chcę otrzymywać e-maile z zapowiedziami nowych funkcji, sugestiami na temat poprawy wydajności, ankietami dotyczącymi opinii użytkowników oraz ofertami specjalnymi.
- Potwierdzam, że będę wykorzystywać wszelkie usługi i związane z nimi interfejsy API zgodnie z obowiązującymi Warunkami korzystania z usług.**
- Wybór opcji:** 'Nie' jest wybrany dla pytania "Chcę otrzymywać e-maile z zapowiedziami nowych funkcji, sugestiami na temat poprawy wydajności, ankietami dotyczącymi opinii użytkowników oraz ofertami specjalnymi."
- Wybór opcji:** 'Tak' jest wybrany dla pytania "Potwierdzam, że będę wykorzystywać wszelkie usługi i związane z nimi interfejsy API zgodnie z obowiązującymi Warunkami korzystania z usług."

At the bottom of the dialog box are two buttons: 'ANULUJ' (Cancel) and 'UTWÓRZ' (Create).

# Tworzenie aplikacji przez interfejs WWW

The screenshot shows the Google Cloud Platform App Engine dashboard. At the top, there's a blue header bar with the text "Google Cloud Platform Project123" and a search icon. Below the header, the navigation bar shows "App Engine". The main content area has a title "Witamy w App Engine" (Welcome to App Engine) and a subtitle "Doskonała platforma do tworzenia aplikacji sieciowych i mobilnych z automatycznym skalowaniem" (A perfect platform for creating network and mobile applications with automatic scaling). It features a large blue box titled "Twoja pierwsza aplikacja" (Your first application) with text about creating a "Hello World" app. Below this are logos for various supported languages: Node.js, Java™, Python, PHP, Go, and Ruby. To the right, there's a section titled "Dokumentacja App Engine" (App Engine documentation) with links to "Przeglądaj dokumentację" (View documentation) and "Pobierz SDK Google App Engine" (Download Google App Engine SDK).

# Wybór lokalizacji

App Engine Your first app with Java

1 Wybierz Lokalizację      2 Wdrażanie

**W której lokalizacji chcesz udostępniać tę aplikację?**

Twoja aplikacja będzie udostępniana w wybranej lokalizacji. Każdy będzie mógł używać tej aplikacji, ale użytkownicy znajdujący się bliżej wybranego regionu będą pracować z mniejszymi opóźnieniami. Lokalizacji tego projektu nie można później zmienić.

A world map with several regions highlighted by blue hexagonal icons with a red dot. Labeled regions include: AMERYKA PÓŁNOCNA (North America), EUROPA (Europe), AZJA (Asia), AFRYKA (Africa), AMERYKA POŁUDNIOWA (South America), and OCEANIA (Oceania). The map also shows major oceans: Ocean Atlantycki, Ocean Spokojny, Ocean Indyjski, and the Pacific Ocean. A dashed line indicates the international date line.

Dane do Mapy ©2017 | Warunki korzystania z programu

Wybierz lokalizację

us-central

W której lokalizacji chcesz udostępniać tę aplikację?

Twoja aplikacja będzie udostępniana w wybranej lokalizacji. Każdy będzie mógł używać tej aplikacji, ale użytkownicy znajdujący się bliżej wybranego regionu będą pracować z mniejszymi opóźnieniami. Lokalizacji tego projektu nie można później zmienić.



Wybierz lokalizację

This region does not support the App Engine flexible environment. ↗

europe-west

Next

# GAE Flexible Environment (1)

Google Cloud Platform

Szukaj Konsola

Why Google Products Solutions Launcher Pricing Customers Documentation Support Try It Free Contact Sales

Standard Environment

Flexible Environment

About the Flexible Environment

Python Java Node.js Go Ruby Custom Runtimes Known Issues

Resources

Choosing an Environment Install the SDK for App Engine App Engine Locations Pricing and Quotas Articles Support Service Level Agreement

## App Engine Flexible Environment

Spis treści Get started

This is a Beta release of the App Engine flexible environment. It is not covered by any SLA or deprecation policy and the implementation may change, possibly in backward-incompatible ways. It is not recommended for production use.

For information about computing and hosting options, see [Computing and hosting services](#).

Note that the flexible environment was initially named "Managed VMs."

Welcome to the App Engine flexible environment. App Engine allows developers to focus on doing what they do best, writing code. Based on [Google Compute Engine](#), the App Engine flexible environment automatically scales your app up and down while balancing the load. Microservices, authorization, SQL and noSQL databases, traffic splitting, logging, versioning, security scanning, and content delivery networks are all supported natively. In addition, the App Engine flexible environment allows you to customize your runtime and even the operating system of your virtual machine using Dockerfiles.

# GAE Flexible Environment (2)

- **Runtimes** - Elastyczne środowisko obejmuje natywne wsparcie dla Java 8/Servlet 3.1 / Jetty 9, Python 2.7 i Python 3.5, Node.js, Ruby and Go. Programiści mogą dostosować te instancje lub dostarczać swoje własne poprzez przygotowanie niestandardowych obrazów Docker pochodzących od społeczności open source.
- **Dostosowanie infrastruktury** - Ponieważ instancje VM w elastycznym środowisku są maszynami wirtualnymi Google Compute Engine, można skorzystać z bibliotek niestandardowych, korzystać z SSH do debugowania i wdrażania własnych obrazów Docker.
- **Wydajność** – Istnieje możliwość ze skorzystania z szerokiej gamy konfiguracji procesora i pamięci. Można określić, ile mocy obliczeniowej procesora i pamięci każda instancja aplikacji potrzebuje i Elastyczne Środowisko AE zarezerwuje niezbędne zasoby.

# GAE Flexible Environment (3)

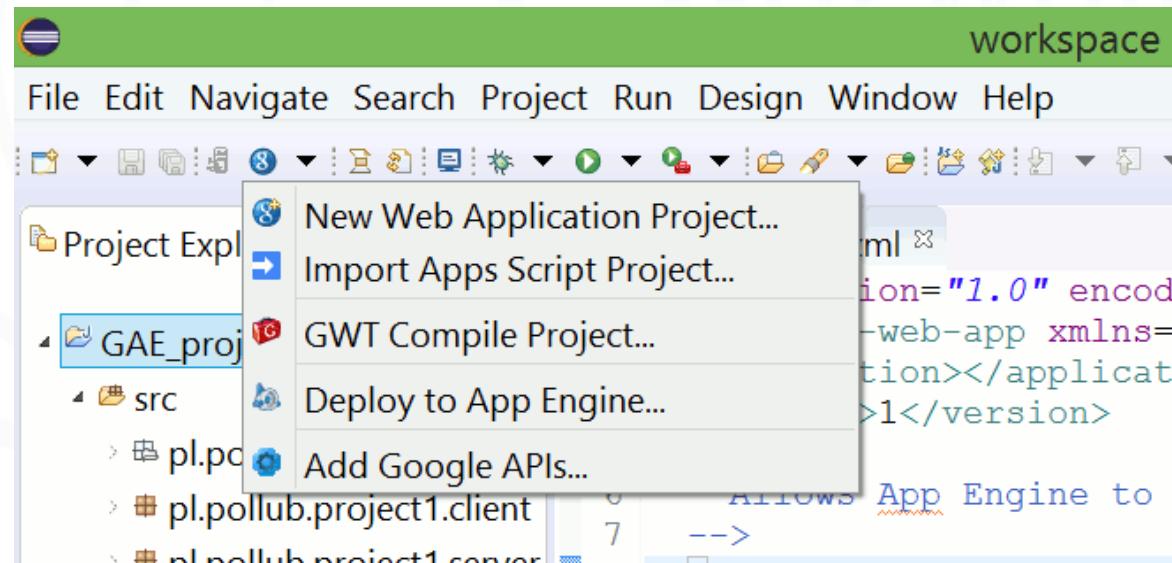
- App Engine zarządza twoimi maszynami wirtualnymi zapewniając że:
  - Instancje są sprawdzane co do zdrowia, reperowane, jeśli to konieczne, i podlegają kolokacji z innymi serwisami w ramach projektu.
  - Aktualizacje krytyczne, wstecznie kompatybilne są automatycznie instalowane w bazowym systemie operacyjnym.
  - Instancje VM są automatycznie rozmieszczane w zależności od regionu geograficznego, zgodnie z ustawieniami w swoim projekcie. Usługi zarządzania Google zapewniają, że wszystkie instancje VM projektu są tak rozmieszczone aby uzyskać optymalną wydajność.

# GAE Flexible Environment (4)

- App Engine zarządza twoimi maszynami wirtualnymi zapewniając że:
  - Instancje VM są restartowane cotygodniowo. Podczas ponownego uruchamiania usługi zarządzania Google zastosuje wszelkie niezbędne aktualizacje systemu operacyjnego i zabezpieczeń.
  - Zawsze masz dostępu do katalogu głównego instancji Compute Engine. Dostęp SSH do instancji VM w Elastycznym Środowisku GAE jest domyślnie wyłączony. Można włączyć dostęp do konta root instancji VM swojej aplikacji.

# Upublicznienie aplikacji z Eclipse

- Bór przycisk Google w Eclipse → Deploy to App Engine ...



# Gdzie rozpocząć

- Przykłady Google App Engine Hello World
- JAVA:
  - <http://www.mkyong.com/google-app-engine/google-app-engine-hello-world-example-using-eclipse/>
- Python:
  - <http://www.mkyong.com/google-app-engine/google-app-engine-python-hello-world-example-using-eclipse/>
- Więcej informacji:
  - <https://developers.google.com/appengine/>

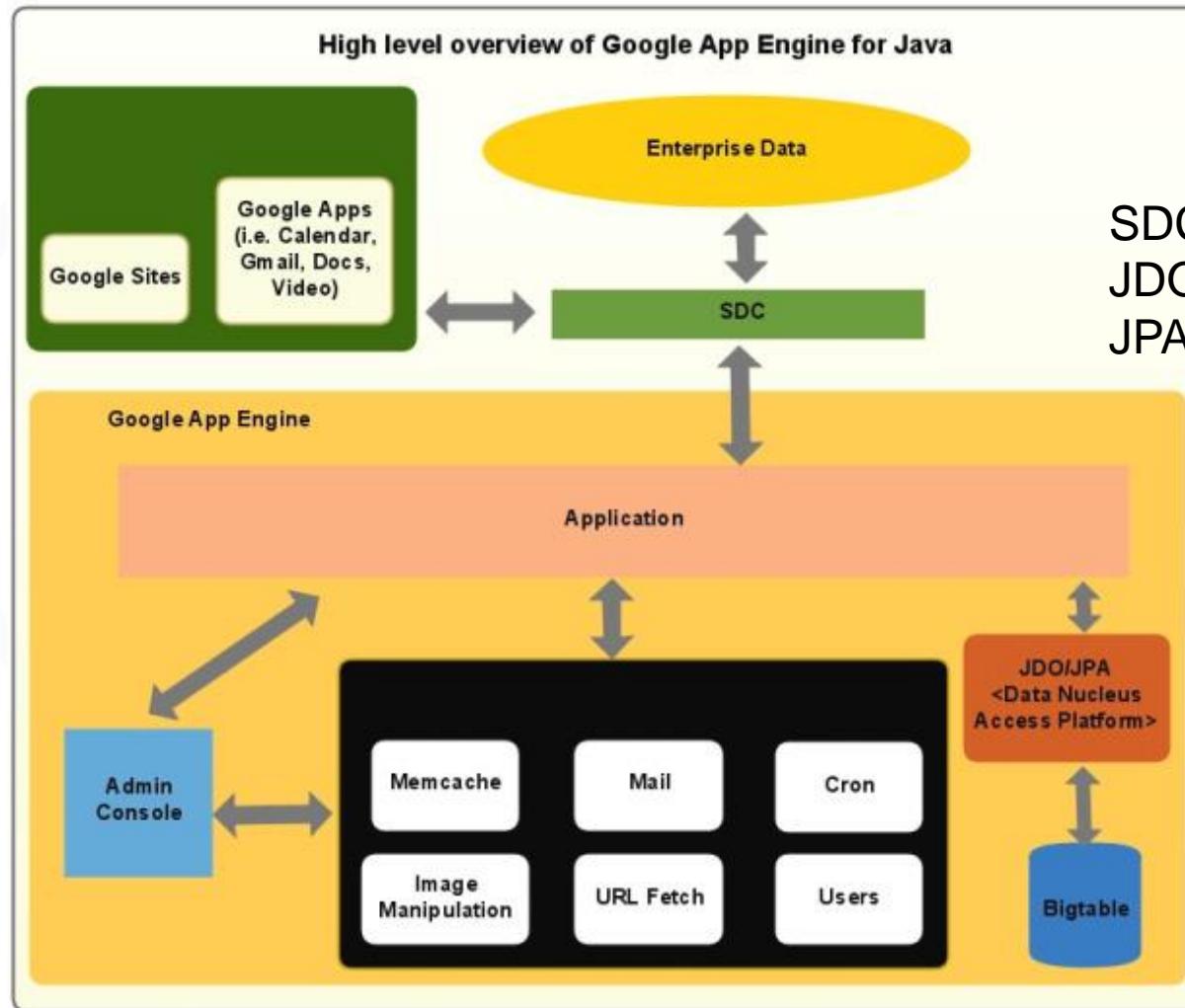
# App Engine robi jedną rzecz dobrze

- App Engine obsługuje zapytania HTTP(S), nic więcej:
  - myśląc RPC: request in, processing, response out
  - działa świetnie dla zastosowań web i AJAX jak również dla innych usług
- Konfiguracja aplikacji jest prosta
  - Nie jest wymagane dostrajanie wydajności
- Wszystko jest zbudowane w skali
  - “nieskończona” liczba aplikacji, zapytań/sekundę, pojemności składowania danych
  - proste API

# Serwisy

- URLFetch – wydobądź zasoby/serwisy web
- Images – manipulacja grafiką: rozmiar, obrót, odbicie, przycinanie
- Google Accounts
- Mail
- XMPP – natychmiastowe przesyłanie wiadomości
- Task Queue – message queue; pozwala na integracje z aplikacjami innymi niż Google API's
- Datastore – zarządzanie obiektami danych
- Blobstore – duże pliki, znacznie większe od obiektów danych, używają par <klucz, obiekt> do dostępu

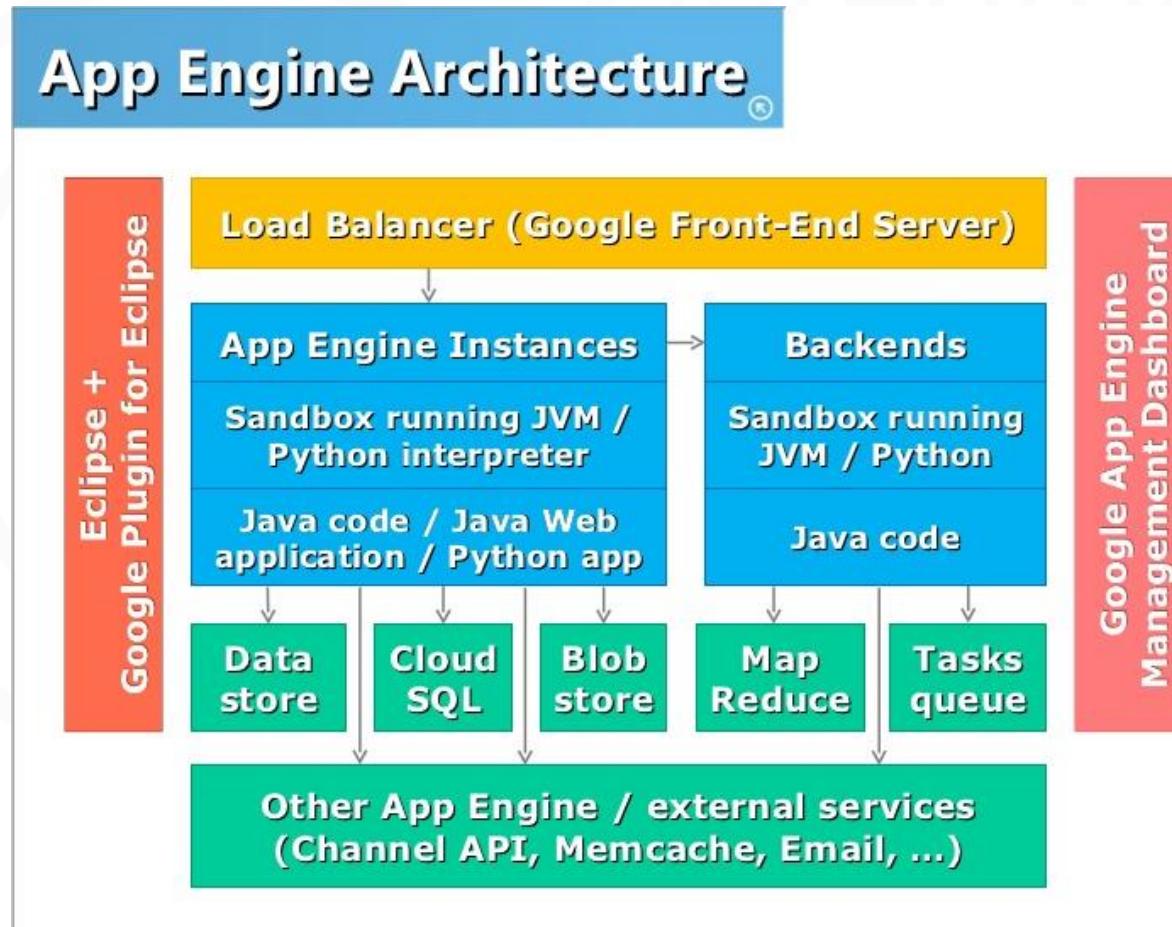
# Architektura App Engine (java)



SDC:secure data connector  
JDO: java data object  
JPA: java persistent API

źródło: [www.byteonic.com/2009/why-java-is-a-better-than-using-python-on-google-app-engine](http://www.byteonic.com/2009/why-java-is-a-better-than-using-python-on-google-app-engine)

# Architektura GAE w ujęciu java



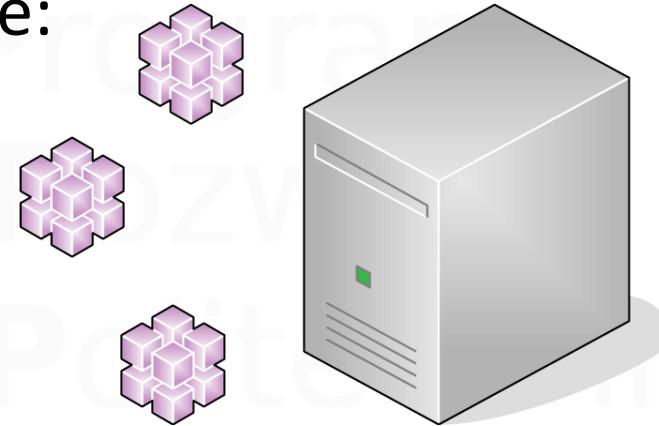
źródło: <http://academy.telerik.com>

# Java w GAE

- Java Servlets, JSPs na silniku Google AppEngine
- Programista zapewnia klasy servlet, JavaServer Pages (JSPs), pliki statyczne i pliki z danymi, jak również deskryptor rozmieszczenia (plik web.xml) i inne pliki konfiguracyjne, w standardowej strukturze katalogów pliku WAR.
- App Engine obsługuje zapytania poprzez wywołanie servletów tak jak zostało to podane w deskryptorze rozmieszczenia.

# Sposób implementacji skalowania w GAE prowadzi do ograniczeń kodzie

- Aplikacje mało wymagające:



- Aplikacje wymagające:



# Sposób implementacji skalowania w GAE prowadzi do ograniczeń kodzie

- Bezstanowe API są łatwe do replikacji
- Składowanie danych zbudowane na Bigtable, zaprojektowane do łatwego skalowania
  - Abstrakcja jako obudowa Bigtable
  - Skalowalność wpływa na API
    - Brak złączeń
    - Rekomendacje: denormalizować schemat, wstępnie przetwarzając złączenia

# Restrykcje w Java uruchamianej w GAE – nowy sposób uzyskania skalowalności

- Aby pozwolić GAE na dystrybucję zapytań dla aplikacji pomiędzy wieloma serwerami web, i aby zapobiec interferencji pomiędzy nimi, aplikacje są uruchamiane w piaskownicy.
- JVM jest uruchomione w zabezpieczonej piaskownicy aby izolować dla usług i bezpieczeństwa.
- Piaskownica zapewnia, że aplikacje mogą **wykonywać tylko czynności, które nie kolidują z wydajnością i skalowalnością innych aplikacji**.

# Restrykcje Java w GAE

- **Aplikacja nie może wywoływać wątków, zapisywać danych do lokalnego systemu plików lub dokonywać dowolnych połączeń sieciowych** (nie ma pozwolenia na składowanie danych w systemie lokalnym gdy kod powstaje w środowisku rozproszonym – generuje to problemy, należy użyć w zamian datastore).
  - Jakkolwiek, aplikacje mogą używać serwisu **URL Fetch** aby uzyskać dostęp do zasobów WEB oraz komunikować się z innymi hostami za pomocą protokołów HTTP i HTTPS. Aplikacje Java mogą używać klasy **java.net.URLConnection** i podobnych do tego aby korzystać z serwisu.
- **Aplikacja nie może używać Java Native Interface lub innego kodu natywnego.**

# Czego nie można w GAE

- **Zapisywać do systemu plików.** ROZWIĄZANIE - Aplikacje muszą używać App Engine datastore dla składowania bieżących danych. Odczyt z systemu plików jest dozwolony, wszystkie wysłane pliki aplikacji są dostępne.
- **Otwarcie gniazda lub bezpośredni dostęp do hosta.** ROZWIĄZANIE - Aplikacja może użyć serwisu App Engine URL fetch aby obsłużyć zapytania HTTP lub HTTPS do innych hostów na portach 80 lub 443.
- **Wywołać podproces lub nowy wątek.** ZASTRZEZENIE - Zapytanie web do aplikacji musi być obsłużone w jednym procesie w przeciągu kilku sekund. Procesy, których odpowiedź zajmuje bardzo dużo czasu są zamkane, aby uniknąć przeciążenia serwera.
- **Tworzyć innych rodzajów wywołań systemowych.**

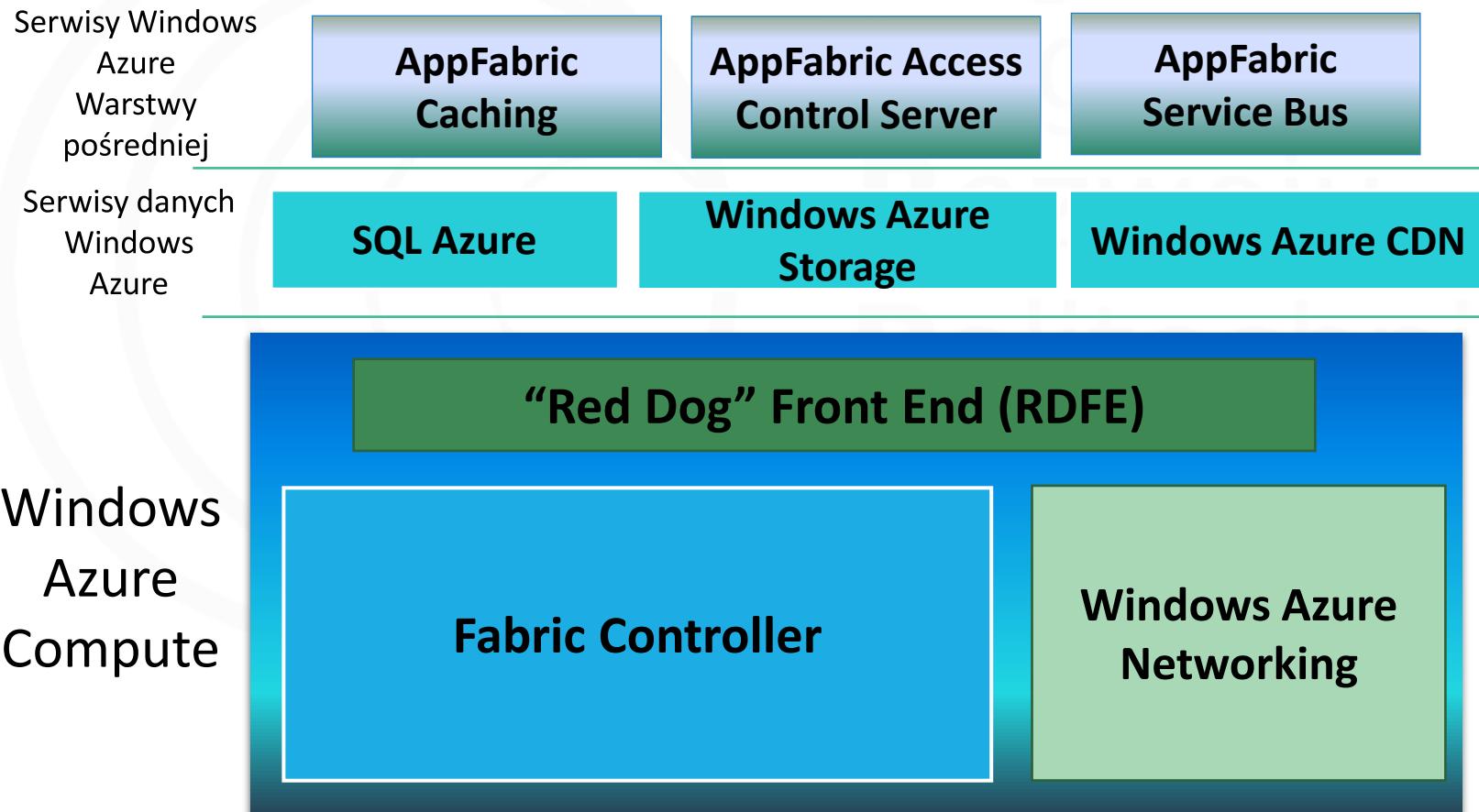
# Windows Azure



- Windows Azure jest systemem dla centrów danych
  - Obsługuje zarządzanie zasobami, aprowizację i monitoring
  - Zarządza cyklem życia aplikacji
  - Pozwala programistom skupić się na logice biznesowej
- Zapewnia wspólne elementy na rzecz aplikacji rozproszonych
  - Niezawodne kolejkowanie, proste struktury danych, bazę SQL
  - Usługi aplikacyjne, takie jak kontrola dostępu, buforowanie i łączność

# Platforma Windows Azure

## Aplikacje Windows Azure



# Windows Azure

- **Fabric Controller:** Zestaw zmodyfikowanych wirtualnych obrazów Windows Server 2008 działających w całej platformie Azure, które kontrolują aprowizację i zarządzanie
- **Fault Domain:** Zestaw zasobów w obrębie centrum danych Azure, które są uważane za nie odporne na uszkodzenia i traktowane jako pojedyncze urządzenie, tak jak pojedyncza szafa rackowa serwerów. Usługa domyślnie dzieli wirtualne instancje na co najmniej dwie domeny awarii (Fault Domain).
- **Role (Rola):** nazwa Microsoft's dla specyficznej konfiguracji magazyny wirtualnej Azure. Terminologia pochodzi z Hyper-V.
- **Service (Usługa):** Azure pozwala użytkownikom na uruchomienie usług, które są wstępnie prekonfigurowane , np. takie jak Rola Web lub Worker. Usługa jest zestawem instancji, takim że wszystkie są regulowane przez parametry i polityki.

# Windows Azure – Red Dog Front End

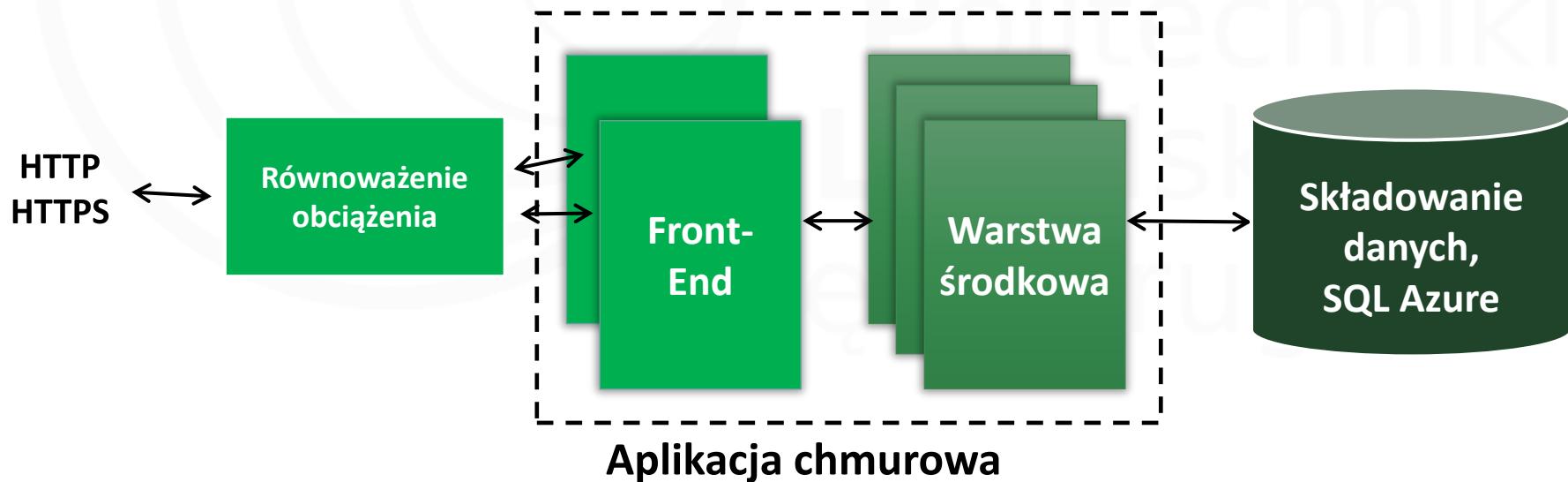
- RDFE służy jako **front-end** dla wszystkich usług Windows Azure
  - zarządzanie abonamentem
  - rozliczenia
  - dostęp użytkowników
  - zarządzanie usługami
- **RDFE** jest odpowiedzialny za wybór klastrów do rozpowszechnienia usług i kont magazynowych
  - Pierwszy region centrów danych
  - Następnie grupa powinowactwa lub równoważenia obciążenia klastra
  - Znormalizowany VIP (virtual IP address) i wykorzystanie rdzenia

# Aplikacja Chmurowa

- Marketing chce wykryć naruszenia wizerunkowe
  - Jako wejście pobierany jest pliku PowerPoint i należy go przeskanować pod kątem "łamania praw do wizerunku" (wykorzystanie "Azure" bez "Windows" lub prefiks "SQL")
- Wymagania:
  - Duża dostępność
  - Strona internetowa IIS / MVC2
  - Skalowalni agenci skanowania naruszenia

# Wielo-Warstwowa Aplikacja Chmurowa

- Aplikacja chmurowa składa się z różnych elementów:
  - Front-End: np. równoważenie obciążenia bezstanowych serwerów internetowych
  - Usługa warstwy średniej: np. przetwarzanie zamówień, kodowanie
  - Składowanie danych: np. tabele SQL lub plików
  - Wielokrotne instancje każdego dla skalowalności i dostępności



# Model Serwisów Windows Azure

- Aplikacja Windows Azure application jest nazywana „usługą”
  - Informacja o definicji
  - Informacja o konfiguracji
  - Zdefiniowana co najmniej jedna rola
- Role są takie jak biblioteki DLL w procesie usług
  - Kolekcja kodu z punktem wejścia, który działa we własnej maszynie wirtualnej
- Obliczeniowa SLA Windows Azure wymaga dwóch wystąpień każdej roli
  - 99,95% dla połączeń dwu instancji
  - Osiągana przy aktualizacji i domenach kolizyjnych

# Zawartość Roli

- Definicja:
  - nazwa roli
  - typ roli
  - rozmiar VM  
(np. mały, średni, itp.)
  - węzły końcowe sieci
- Kod:
  - Rola Web/Worker:  
Hostowany DLL  
i inne pliki wykonawcze
  - Rola VM: VHD
- Konfiguracja:
  - liczba instancji
  - liczba domen aktualizacji i kolizyjnych

## Serwis chmurowy

### Rola: Front-End

#### Definicja

Typ: Web  
Rozm. VM: Small

Endpoints:  
External-1

#### Konfiguracja

Instancje: 2  
Domeny aktual.: 2  
Domeny kol.: 2

### Rola: Middle-Tier

#### Definicja

Typ: Worker  
Rozm. VM: Large

Endpoints:  
Internal-1

#### Konfiguracja

Instancje: 3  
Domen aktual.: 2  
Domeny kol.: 2

# Typy roli

- Obecnie występują trzy role:
  - Web Role: IIS7 oraz ASP.NET w systemie Windows Azure
  - Worker Role: dowolny kod w systemie dostarczonym przez Windows Azure
  - VM Role: załadowany VHD z systemem gościa dostarczonym przez klienta
- Rola VM a maszyna wirtualna
  - Rola nie jest maszyną wirtualną ponieważ jest bezstanowa
  - Dobra dla:
    - długiej instalacji (5 min. i dłużej)
    - ręczna instalacja/konfiguracja
    - „delikatną” instalację/konfigurację

# Pliki Modelu Serwisów

- Definicja serwisu jest umieszczona w **ServiceDefinition.csdef**
- Konfiguracja serwisu umieszczona jest w **ServiceConfiguration.cscfg**
- Program CSPack pakuje z użyciem ZIP pliki binarne serwisu i definicje do pakietu **service.cscfg**

Name	Type	Size
ServiceConfiguration	CSCFG File	3 KB
Thumbnails	Service Package file	2,972 KB

```
<?xml version="1.0" encoding="utf-8"?>
<ServiceDefinition name="Thumbnails" xmlns="http://schemas.micros...
<WorkerRole name="Thumbnails_WorkerRole">
  <ConfigurationSettings>
    <Setting name="DataConnectionString" />
    <Setting name="DiagnosticsConnectionString" />
  </ConfigurationSettings>
</WorkerRole>
<WebRole name="Thumbnails_WebRole">
  <InputEndpoints>
    <!-- Must use port 80 for http and port 443 for https when
        <InputEndpoint name="HttpIn" protocol="http" port="80" />
    </InputEndpoints>
    <ConfigurationSettings>
      <Setting name="DataConnectionString" />
      <Setting name="DiagnosticsConnectionString" />
    </ConfigurationSettings>
  </WebRole>
</ServiceDefinition>
```

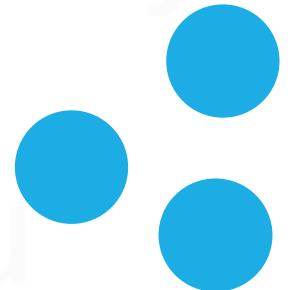
```
<?xml version="1.0"?>
<ServiceConfiguration serviceName="Thumbnails" xmlns="ht...
<Role name="Thumbnails_WorkerRole">
  <Instances count="2" />
  <ConfigurationSettings>
    <!-- Add your storage account information and unc...
      <Setting name="DataConnectionString" value="Defa...
      <Setting name="DiagnosticsConnectionString" val...
    -->
    <Setting name="DataConnectionString" value="Defa...
    <Setting name="DiagnosticsConnectionString" value...
  </ConfigurationSettings>
</Role>
<Role name="Thumbnails_WebRole">
  <Instances count="1" />
  <ConfigurationSettings>
    <!-- Add your storage account information and unc...
      <Setting name="DataConnectionString" value="Defa...
      <Setting name="DiagnosticsConnectionString" val...
    -->
    <Setting name="DataConnectionString" value="Defa...
    <Setting name="DiagnosticsConnectionString" value...
  </ConfigurationSettings>
</Role>
</ServiceConfiguration>
```

# Podstawy Windows Azure Storage

- Charakterystyka pamięci masowej
  - Trwałość – potrójna replikacja danych
  - Skalowalność – pojemność i przepustowość
  - Wysoka dostępna
- Proste i znajome interfejsy programowania
  - REST (HTTP i HTTPS)
  - .NET dostępne

# Obiekty pamięci masowych

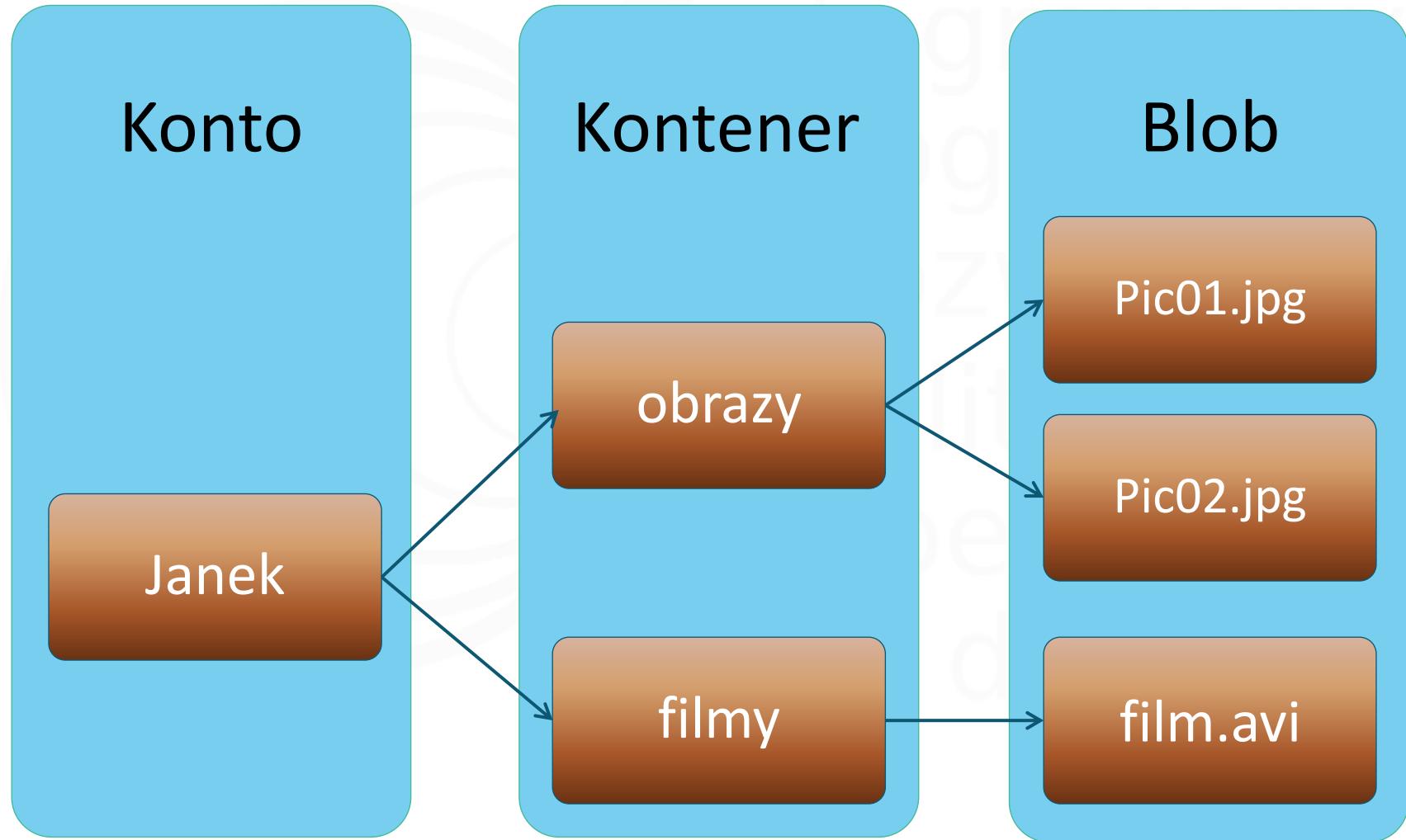
- Bąble - Blobs
  - Zapewniają prosty interfejs do przechowywania nazwanych plików oraz metadanych pliku
- Tabele - Tables
  - Zapewniają przechowywanie w lekko ustalonej strukturze składającej się z grupy jednostek, które zawierają zestaw właściwości
- Kolejki - Queues
  - Zapewnić niezawodne przechowywanie i dostarczanie wiadomości



# Konto Magazynowe i Kontenery Blob

- Konto magazynowe
  - Konto może posiadać wiele kontenerów Blob (bąbelkowych)
- Kontener
  - Kontener jest zestawem wielu „bąbli”
  - Współdzielenie polityk odbywa się na poziomie kontenerów
    - typy Public READ lub Private
  - Przypisanie metadanych do kontenera
    - Metadane to pary <nazwa, klucz>
    - Maksymalnie do 8KB na kontener

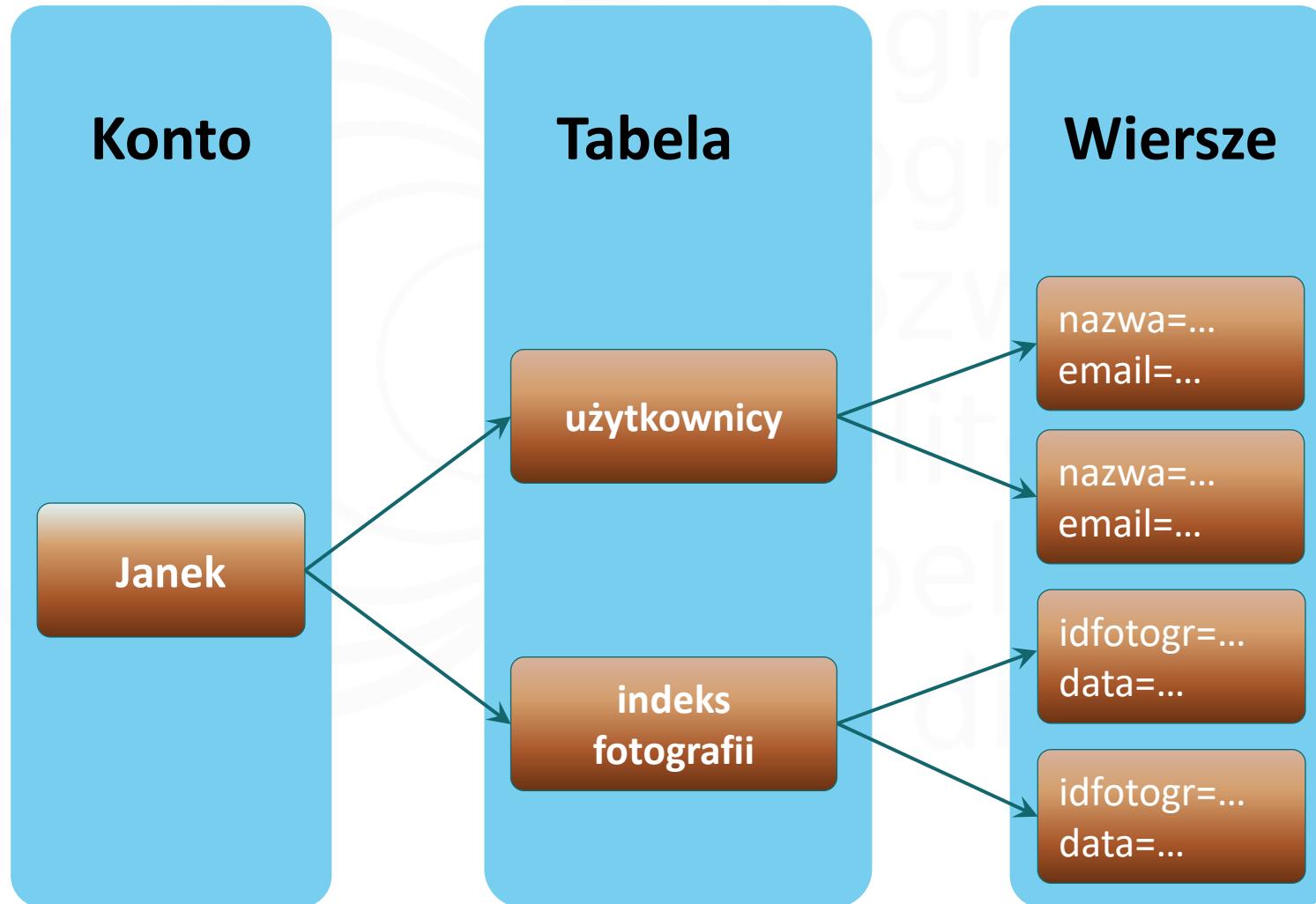
# Koncepcja magazynu typu Blob (bąbelkowy)



# Tablicowy Model Danych

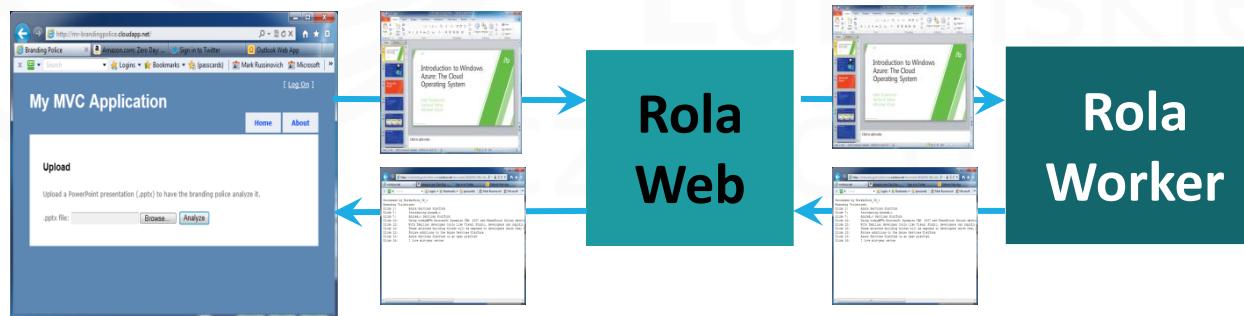
- Tabele
  - Konta przechowywania danych mogą tworzyć wiele tabel
  - klasy .NET oraz LINQ
- Tabela jest zbiorem wierszy
  - wiersze składają się z kolumn
  - Miliardy wierszy i TB-ów danych
- Dwie kluczowe "właściwości", stanowią razem unikalny klucz w tabeli
  - PartitionKey - umożliwia skalowalność
  - RowKey - jednoznacznie identyfikuje wiersz w partycji

# Koncepcja magazynu tabel



# Komunikacja między rolami

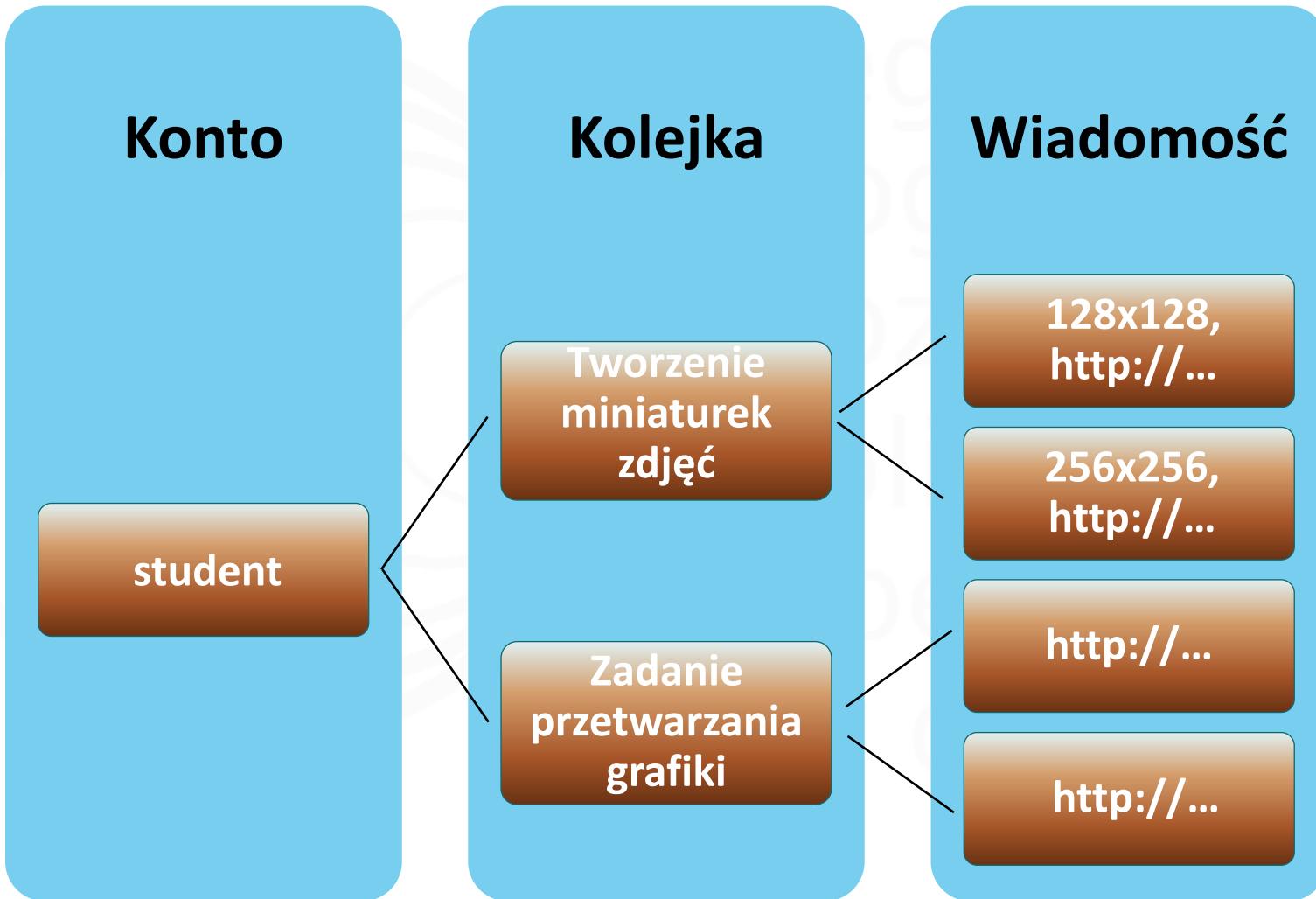
- Mamy kilka rodzajów komunikacji między rolami:
  - Plik PowerPoint wysłany z roli Web do roli Worker
  - Naruszenia dotyczące marki zwracane przez rolę Worker do użytkownika
- Wymagania:
  - Komunikacja musi być asynchroniczna
  - Musi obsługiwać równoczesne wykrywanie naruszeń dla różnych użytkowników
  - Awaria dowolnego węzła musi w najgorszym przypadku spowodować opóźnienie



# Kolejki Windows Azure

- Zapewniają niezawodne dostarczanie wiadomości
  - Prosta, asynchroniczna wysyłka zadań
  - Semantyka programowania zapewnia, że komunikat może zostać przetworzony co najmniej raz
- Kolejki są wysoce dostępne, trwałe i wydajne
  - Maksymalny rozmiar to 64 KB
  - FIFO obsługa generalna, ale nie jest gwarantowana
- Wyciągnięcie elementu z kolejki nie powoduje jego usunięcia
  - Staje się niewidoczny do limitu czasu widoczności
  - Element musi zostać usunięty przed upływem limitu czasu, w przeciwnym razie stanie się widoczny

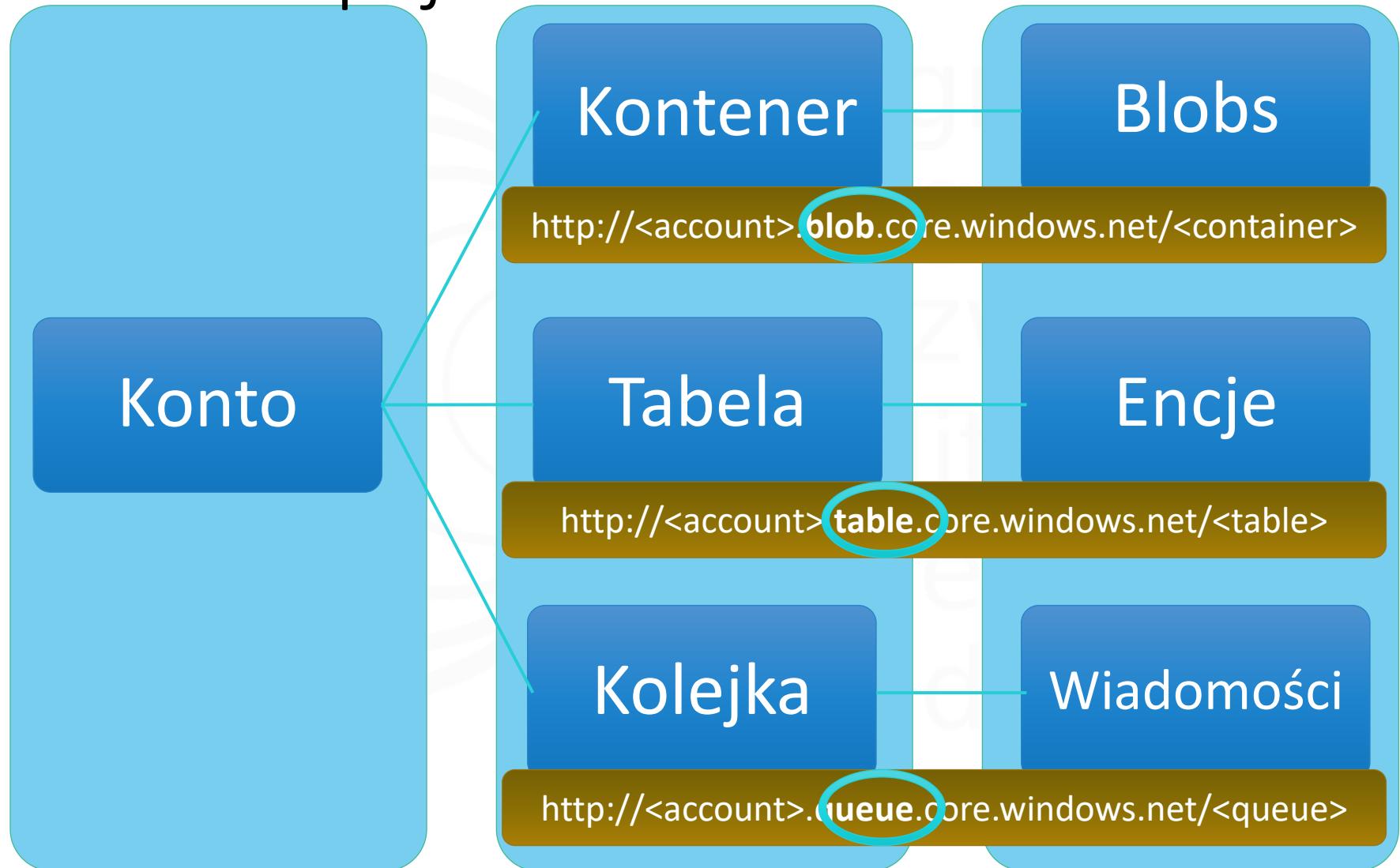
# Koncepcja kolejek składowania



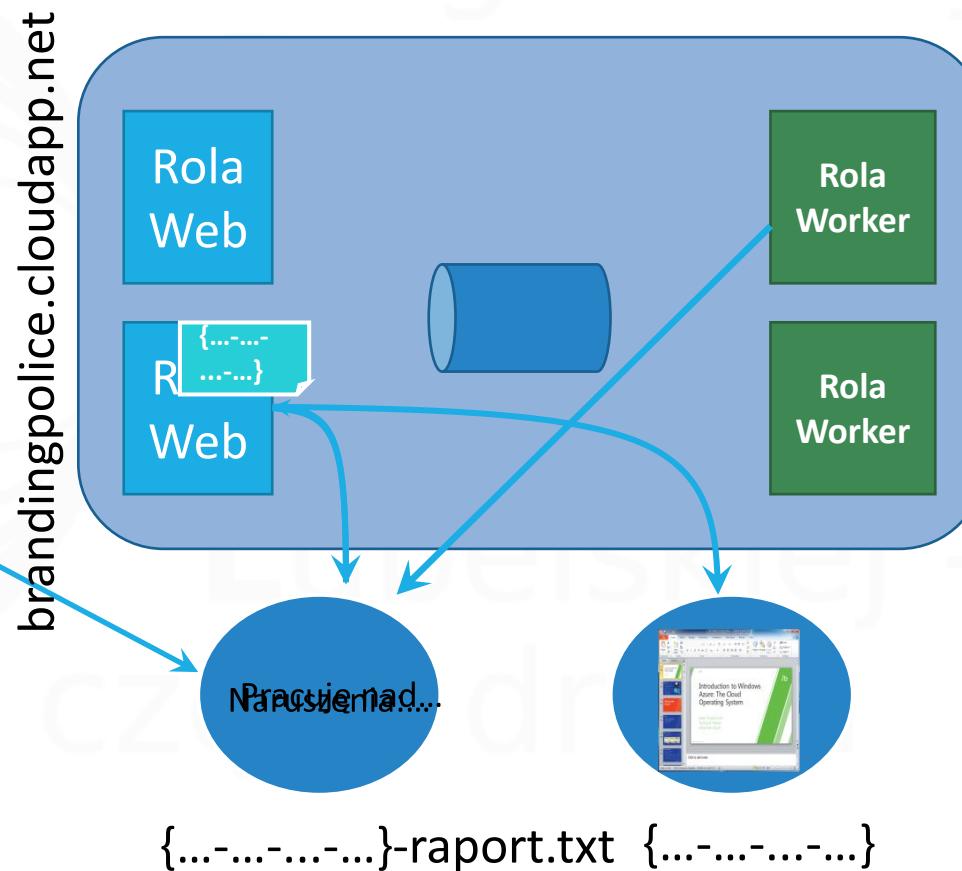
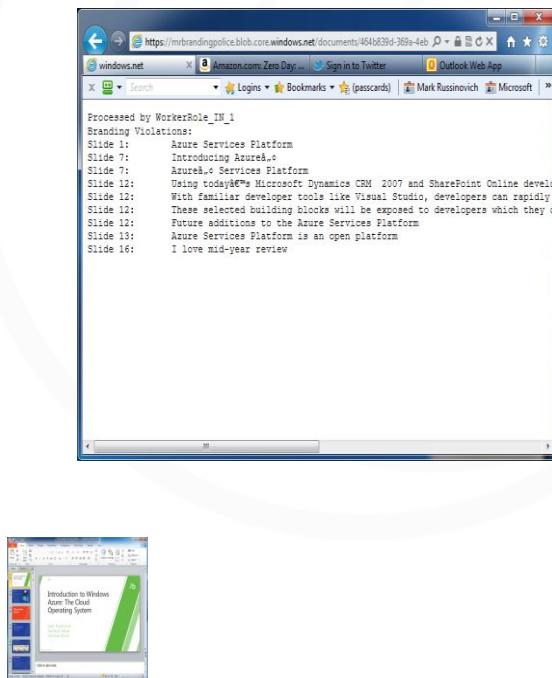
# Wnioski

- Chmura umożliwia samoobsługowe (pay-as-you-go) dostarczanie zasobów aplikacji zgodnie z rzeczywistym użyciem
- Platforma jako usługa polega na zmniejszeniu kosztów zarządzania i operacji
- Windows Azure umożliwia tworzenie i wdrażanie skalowalnych aplikacji o wysokiej dostępności w ciągu kilku minut
- W systemie Windows Azure można wdrożyć kod przy użyciu dowolnego języka lub środowiska wykonawczego Windows
- Gdzie zacząć - Windows Azure Training course
  - <https://docs.microsoft.com/pl-pl/learn/azure/>

# Koncepcja Windows Azure Data

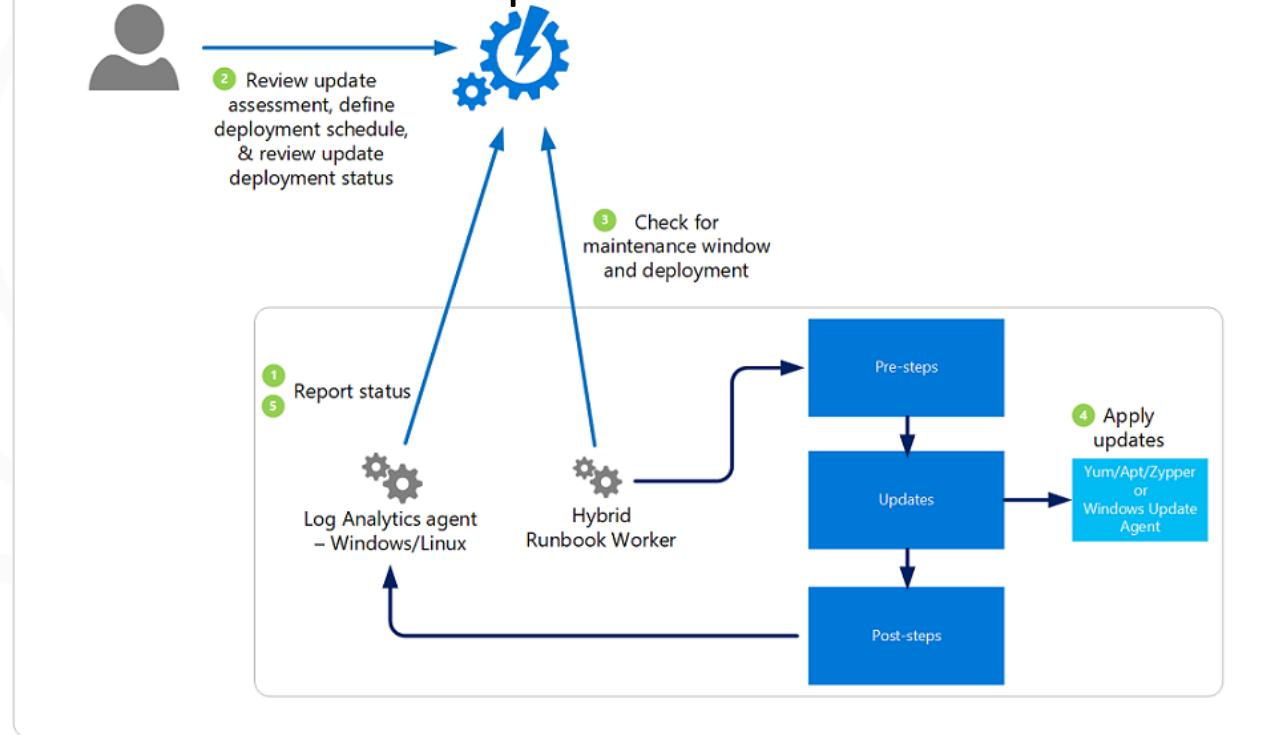


# Przykład naruszeń



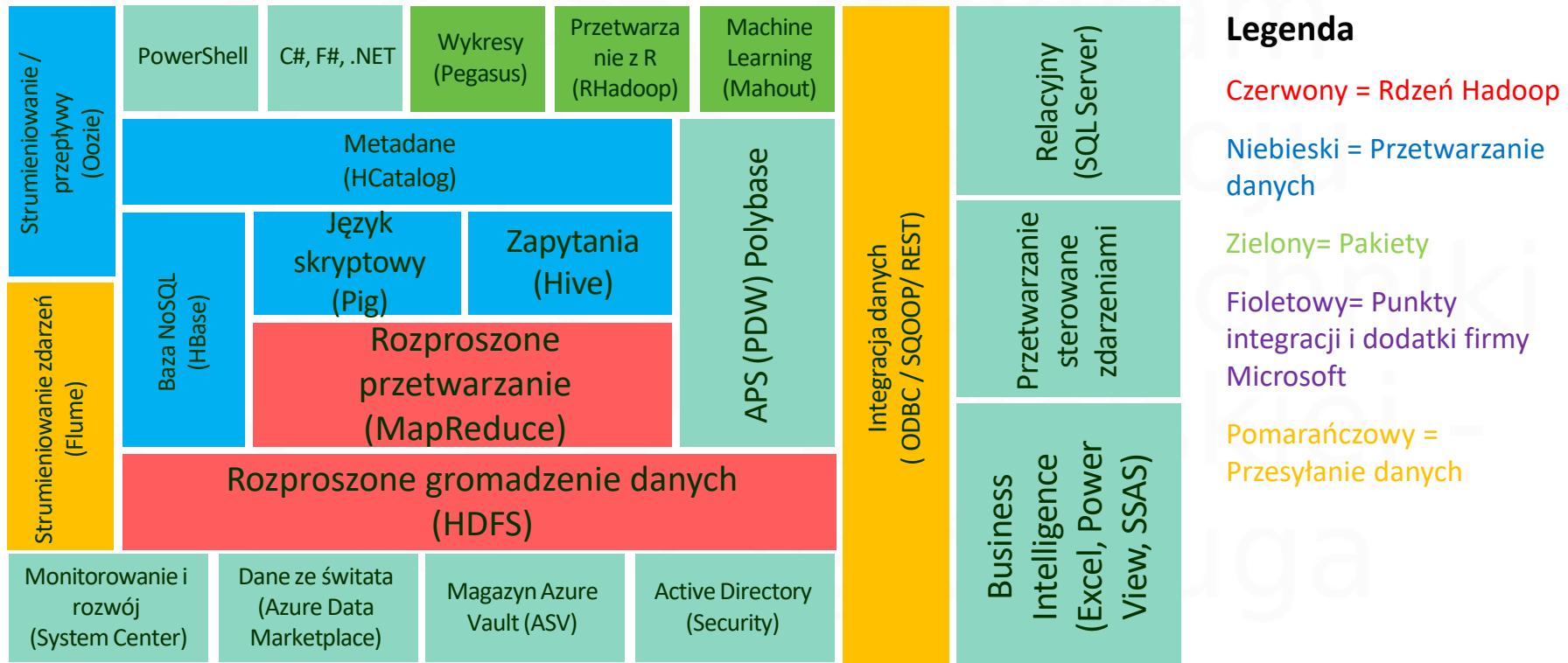
# Aktualizacja usług Windows Azure

- Za pomocą Update Management w usłudze Azure Automation można zarządzać aktualizacjami systemu operacyjnego dla maszyn wirtualnych z systemem Windows i Linux na platformie Azure.



źródło: <https://docs.microsoft.com/en-us/azure/automation/update-management/overview>

# Wstęp do MS Azure Hadoop: ekosystem HDInsight/Hadoop



# Programowanie w HDInsight

- Ponieważ HDInsight jest implementacją opartą na usługach, można natychmiast uzyskać dostęp do narzędzi potrzebnych do programowania w HDInsight / Hadoop

Istniejący ekosystem

- Hive, Pig, Sqoop, Mahout, Cascading, Scalding, Scoobi, Pegasus, itp.

.NET

- C#, F# Map/Reduce, LINQ to Hive, .Net zarządzalni klienci, itp.

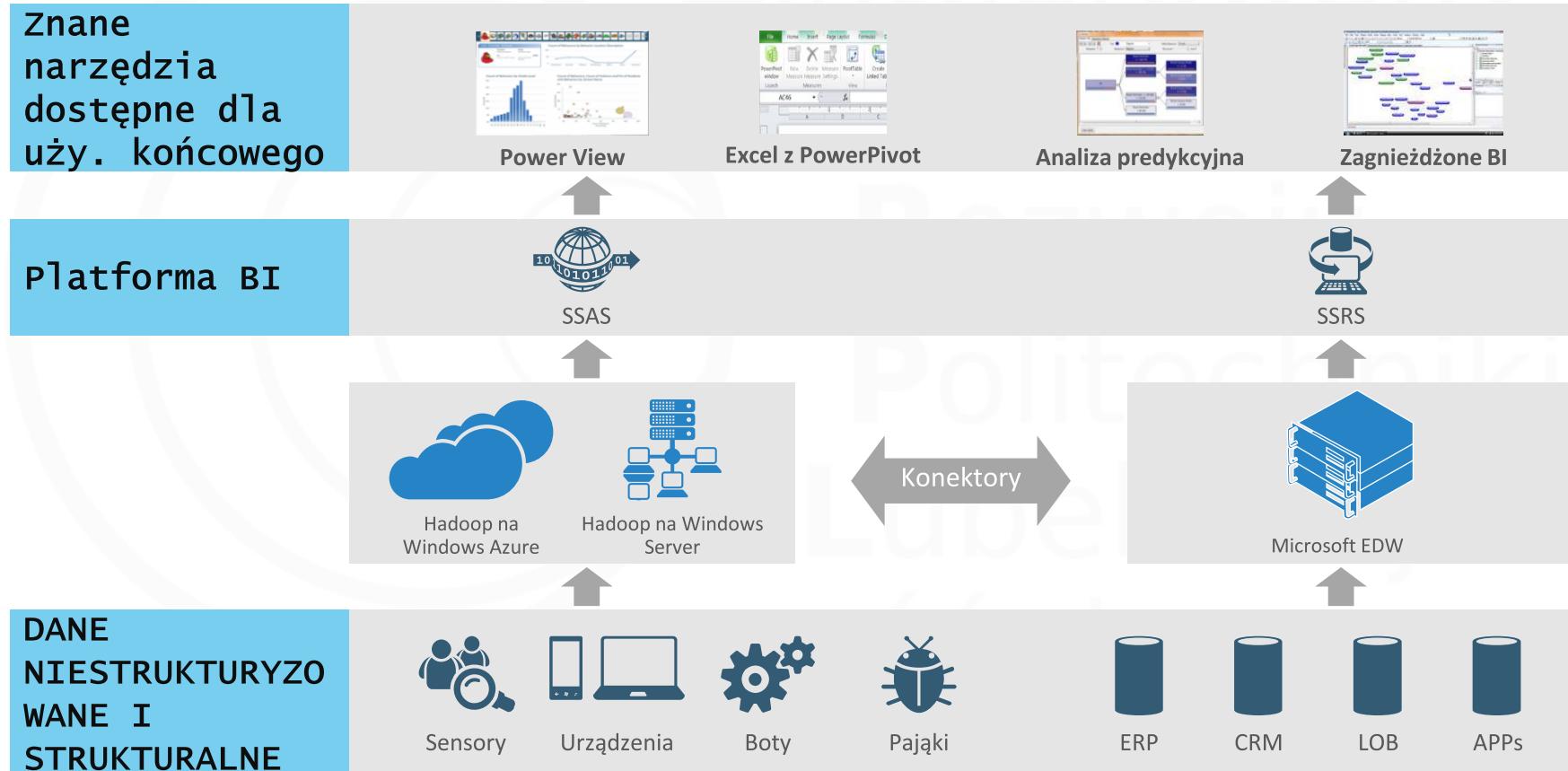
JavaScript

- JavaScript Map/Reduce, Konsola przeglądarkowa, Node.js zarządzalni klienci

DevOps/IT Pros:

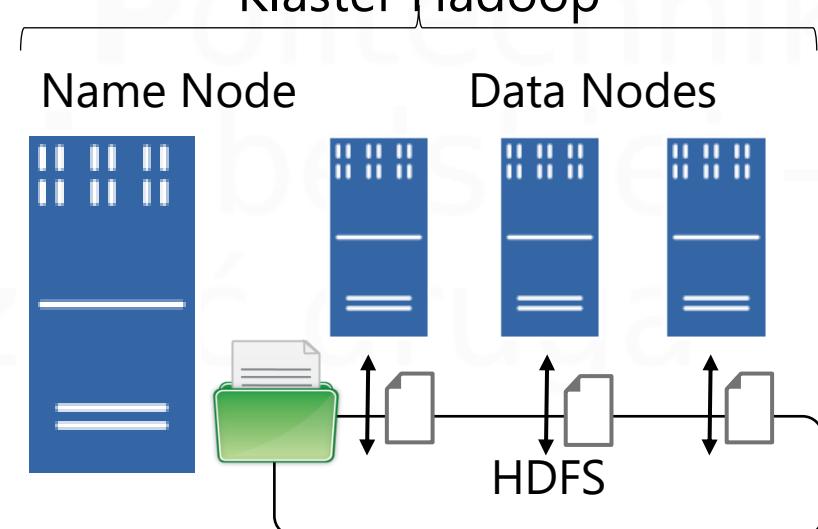
- PowerShell, Narzędzia międzyplatformowe CLI

# Rozwiązania Microsoft Big Data



# Hadoop w MS Azure

- Hadoop
  - Klaster typu open source do rozproszonego przetwarzania danych
  - Dane przetwarzane w rozproszonym systemie plików Hadoop (HDFS)
  - Zarządzanie zasobami jest wykonywane przez YARN
- Powiązane projekty
  - Hive
  - Pig
  - Oozie
  - Sqoop
  - Inne





Materiały zostały opracowane w ramach projektu  
„Zintegrowany Program Rozwoju Politechniki Lubelskiej – część druga”,  
umowa nr **POWR.03.05.00-00-Z060/18-00**  
w ramach Programu Operacyjnego Wiedza Edukacja Rozwój 2014-2020  
współfinansowanego ze środków Europejskiego Funduszu Społecznego

