Skalowalność usług w chmurze

Cloud Computing zapewnia łatwy sposób korzystania i dostępu do dużej puli zwirtualizowanych zasobów (takich jak sprzęt, platformy programistyczne i/lub usługi), które można dynamicznie udostępniać w celu dostosowania do zmiennego obciążenia, co pozwala również na optymalne wykorzystanie zasobów. Ta pula zasobów jest zazwyczaj wykorzystywana w modelu pay-per-use, w którym gwarancje są oferowane za pomocą dostosowanych umów SLA (Vaquero, Rodero-Merino, Caceres i Lindner, 2009). Dlatego mechanizmy zautomatyzowanego udostępniania Cloud Computing mogą pomóc aplikacjom w skalowaniu systemów w górę i w dół w taki sposób, aby zrównoważyć wydajność i zrównoważony rozwój ekonomiczny. Więc co to znaczy skalować? Zasadniczo skalowalność1 można zdefiniować jako „zdolność konkretnego systemu do dopasowania się do problemu w miarę zwiększania się zakresu tego problemu (liczba elementów lub obiektów, rosnący nakład pracy i/lub podatność na rozbudowę)". Na przykład zwiększenie przepustowości systemu poprzez dodanie większej ilości zasobów oprogramowania lub sprzętu, aby poradzić sobie ze zwiększonym obciążeniem (Schlossnagle, 2007; Bondi, 2000). Możliwość skalowania systemu w górę może zależeć od jego konstrukcji, typów struktur danych, algorytmów lub mechanizmów komunikacyjnych wykorzystywanych do implementacji elementów systemu. Charakterystykę różnych typów skalowalności opisuje Bondi (2000), tutaj podsumowujemy kilka istotnych przykładów:

• Skalowalność obciążenia: gdy system ma możliwość dobrego wykorzystania dostępnych zasobów na różnych poziomach obciążenia (tj. unikanie nadmiernych opóźnień, bezproduktywnego zużycia lub rywalizacji). Czynniki, które wpływają na skalowalność obciążenia, mogą być złym wykorzystaniem równoległości, niewłaściwym planowaniem zasobów współdzielonych lub nadmiernymi kosztami ogólnymi. Na przykład serwer sieci Web utrzymuje dobry poziom skalowalności obciążenia, jeśli wydajność systemu jest utrzymywana na akceptowalnym poziomie, gdy liczba wątków wykonujących żądania HTTP wzrasta w szczytowym obciążeniu.

• Skalowalność przestrzeni: system ma zdolność do utrzymywania zużycia zasobów systemowych (tj. pamięci lub przepustowości) na akceptowalnym poziomie, gdy wzrasta obciążenie. Na przykład system operacyjny skaluje się z wdziękiem przy użyciu mechanizmu pamięci wirtualnej, który zamienia nieużywane strony pamięci wirtualnej z pamięci fizycznej na dysk, unikając wyczerpania pamięci fizycznej. Innym przykładem może być wzrost liczby kont użytkowników usługi Web 2.0, takiej jak sieć społecznościowa, z tysięcy do milionów.

• Skalowalność strukturalna: Wdrożenie standardów w systemie pozwala na zwiększenie liczby zarządzanych obiektów lub przynajmniej zrobi to w określonym czasie. Na przykład rozmiar typu danych może mieć wpływ na liczbę elementów, które mogą być reprezentowane (przy użyciu 16-bitowej liczby całkowitej jako identyfikatora jednostki można reprezentować tylko 65 536 jednostek).

Byłoby pożądane, aby możliwości skalowania systemu pozostały zarówno krótko-, jak i długoterminowe; posiadanie krótkoterminowej reaktywności w odpowiedzi na wysoki i niski wskaźnik napływających prac. Równie ważne, jak skalowanie w górę jest zmniejszanie, ma to bezpośredni wpływ na zrównoważony rozwój firmy, zmniejszając koszty eksploatacji nieużywanych zasobów, gdy zmniejsza się obciążenie pracą, unikając nadmiernej alokacji. Czynniki, które mogą poprawić lub zmniejszyć skalowalność, mogą być trudne do zidentyfikowania, a nawet specyficzne dla systemu docelowego. Czasami działania podjęte w celu poprawy jednej z tych zdolności mogą zepsuć inne. Na przykład wprowadzenie algorytmów kompresji w celu poprawy skalowalności przestrzeni (tj. zmniejszenie przepustowości kompresji wiadomości) wpływa na skalowalność obciążenia (tj. zwiększenie wykorzystania procesora podczas kompresji wiadomości). Działania skalowalne można podzielić na:

• Skalowanie w pionie: poprzez dodanie większej mocy (więcej procesorów, pamięci, przepustowości itp.) do sprzętu używanego przez systemy. W ten sposób aplikacje są wdrażane na dużych serwerach z pamięcią współużytkowaną.

• Skalowanie poziome: przez dodanie większej liczby takich samych zasobów oprogramowania lub sprzętu. Na przykład w typowej usłudze dwuwarstwowej więcej węzłów frontonu jest dodawanych (lub zwalnianych), gdy liczba użytkowników i obciążenie pracą wzrasta (zmniejsza się). W ten sposób aplikacje są wdrażane na rozproszonych serwerach.

Skalowalność należy mieć na uwadze od samego początku projektując architekturę systemu. Chociaż odpowiedni czas wprowadzenia produktu na rynek, szybkie prototypowanie lub ukierunkowanie na niewielką liczbę użytkowników mogą wymagać szybkiego rozwoju, architektura rozwiązania powinna uwzględniać skalowalność. Oznacza to, że system może zwiększyć liczbę użytkowników z setek do tysięcy, a nawet milionów lub zwiększyć złożoność. Dzięki temu zminimalizowane zostanie ryzyko awarii i ponownej implementacji systemu. Chmura jest paradygmatem obliczeniowym, którego celem jest, między innymi, ułatwienie sposobu dostarczania usługi, pomagając dostawcom usług poprzez dostarczanie iluzji nieskończonych zasobów bazowych i automatycznej skalowalności. W tym artykule opisano, w jaki sposób Cloud Computing może pomóc w tworzeniu skalowalnych aplikacji poprzez automatyzację procesu świadczenia usług za pomocą chmur IaaS (Infrastructure as a Service) (redukcja kosztów zarządzania i optymalizacja wykorzystania zasobów) oraz dostarczanie frameworków PaaS (Platform as a Service) (ze skalarnymi środowiskami wykonawczymi, blokami konstrukcyjnymi usług i interfejsami API) do tworzenia aplikacji zgodnych z chmurą w modelu oprogramowania jako usługi (SaaS).

Podwaliny

Po wyjaśnieniu tego, co jest obecnie rozumiane przez skalowalność i krótko nakreślonych, skalowalność chmury opiera się na trzech podstawowych filarach:

• Wirtualizacja: zmniejsza złożoność systemów, standaryzując platformę sprzętową, a następnie zmniejszając koszty zarządzania zasobami.

• Współdzielenie zasobów: współdzielenie zasobów obliczeniowych między różnymi aplikacjami i/lub organizacjami pozwoli zoptymalizować ich wykorzystanie, unikając nielicznych lub bezczynnych czasów zajmowania. W tym sensie wirtualizacja pomaga skonsolidować serwer na tej samej maszynie fizycznej.

• Dynamiczne przydzielanie: zasoby powinny być udostępniane na żądanie, powinny być również automatycznie rekonfigurowane w locie. Dynamiczne przydzielanie oznacza potrzebę monitorowania wydajności usług i automatyzacji decyzji i działań w odpowiedzi na rosnące/zmniejszające się obciążenie pracą.

W tej sekcji przeanalizujemy ewolucję usług technologii informacyjnej (IT) od komputerów mainframe do chmury, co wyjaśnia, w jaki sposób rozwiązano problem skalowalności, a także niektóre techniki dynamicznej alokacji zasobów, które mogą pomóc we wdrożeniu automatycznej skalowalności.

Historia usług IT dla przedsiębiorstw

Od lat 60. systemy mainframe są obecne w back-office firm w finansach, opiece zdrowotnej, ubezpieczeniach, administracji i innych przedsiębiorstwach publicznych i prywatnych. Rozpoczynając przetwarzanie zadań wsadowych, wprowadzonych za pomocą kart dziurkowanych, taśm papierowych lub taśm magnetycznych, w ciągu ostatnich dziesięcioleci firma rozwinęła się, dodając interaktywne

terminale i obsługując wiele instancji systemów operacyjnych (OS) obsługiwanych przez maszyny wirtualne. Komputery typu mainframe (Ebbers et al., 2009) to scentralizowane komputery przeznaczone do szybkiej obsługi i przetwarzania bardzo dużych ilości danych przy zachowaniu wysokiej niezawodności, dostępności i łatwości serwisowania. Komputery mainframe skalują się w pionie, dodając więcej zasobów obliczeniowych, pamięci masowej lub łączności. Dzięki ciągłej kompatybilności sprzętu i systemów operacyjnych, aktualizacje systemu operacyjnego, rodziny architektury lub modelu pozwalają na skalowanie systemów bez konieczności zmiany aplikacji, które na nim działają. Aplikacje zdecentralizowane, a w szczególności architektury klient/serwer (Orfali, Harkey i Edwards, 1996), zostały zaimplementowane pod koniec lat 80-tych jako alternatywa dla komputerów mainframe. Aplikacja składa się z autonomicznych jednostek obliczeniowych z własną pamięcią i komunikuje się przez sieć przy użyciu protokołów przekazywania komunikatów (takich jak Remote Procedure Call Protocol, RPC, który sam jest protokołem przekazywania komunikatów). Serwer zapewnia zestaw funkcji do jednego z wielu klientów, którzy wywołują żądania takich funkcji. Systemy rozproszone mają pewne zalety w porównaniu z komputerami typu mainframe:

• Redukcja kosztów: cena komputera typu mainframe wynosiła 2-3 mln USD (obecnie kosztuje około 1 mln USD). Aplikacje średniej wielkości można wdrożyć za zaledwie kilka tysięcy dolarów.

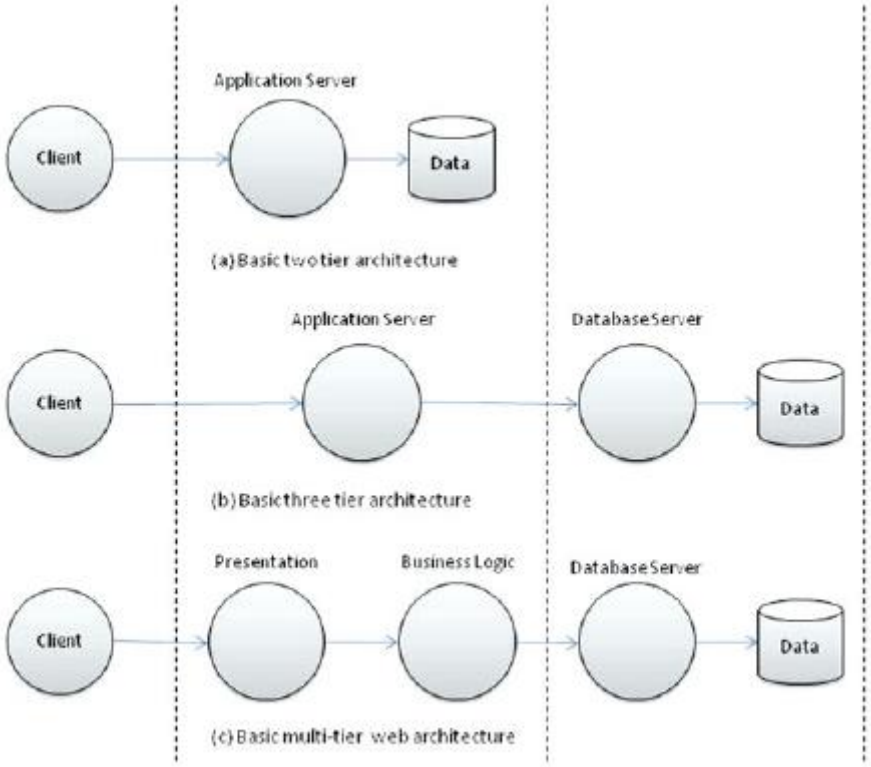
• Elastyczność: Węzły serwerów, zwykle hostowane na dedykowanych komputerach (głównie UNIX), mogą być rozwijane, konfigurowane i testowane oddzielnie od reszty systemu i podłączane, gdy są gotowe.

• Redukcja opóźnień: węzły serwerów mogą być rozmieszczone w różnych centrach danych, aby być jak najbliżej użytkowników końcowych.

• Usługi interaktywne: Początkowo komputery mainframe były zorientowane na przetwarzanie wsadowe, ale klient/serwer jest głównie interaktywny.

• Nieograniczone dodawanie zasobów: komputery mainframe prezentują limity zależne od platformy podczas dodawania większej liczby zasobów (procesor, dysk, pamięć...). Systemy rozproszone pozwalają na dodanie większej liczby serwerów w celu zwiększenia pojemności całego systemu.

Architektury klient/serwer są podstawowym modelem dla obliczeń sieciowych, takich jak usługi internetowe (web, poczta, ftp, streaming itp.), systemy telekomunikacyjne (IMS, VoIP, IPtv itp.) oraz aplikacje korporacyjne (usługi informacyjne, bazy danych itp.).). Rysunek pokazuje, jak architektury klient/serwer ewoluowały od dwuwarstwowej (klient i serwer hostowane na różnych maszynach) do trójwarstwowej (klient, logika aplikacji i warstwa danych) i wielowarstwowej (klient, warstwa logiki prezentacji, warstwa logiki biznesowej i warstwa danych).



Ponieważ aplikacje stają się coraz bardziej złożone, a większość procesów biznesowych została zautomatyzowana, centra danych zajmują coraz więcej przestrzeni fizycznej. Systemy rozproszone mogą skalować się w pionie (dodawanie większej ilości zasobów do węzła hosta), w poziomie (dodawanie nowych węzłów tego samego typu) lub w obu (od hostingu wszystkich węzłów usługowych na tym samym serwerze do dystrybucji na serwerach dedykowanych z wykorzystaniem przejrzystości lokalizacji które protokoły komunikacyjne zazwyczaj zapewniają). Skala pozioma może wymagać przeprojektowania aplikacji w celu wprowadzenia równoległości, równoważenia obciążenia itp. Jednak w latach 90-tych systemy rozproszone miały pewne wady w porównaniu z komputerami typu mainframe:

• Niskie wykorzystanie węzłów serwerów: Zazwyczaj węzły serwerów są dedykowane dla pojedynczego węzła aplikacji, ponieważ wymagają one określonego systemu operacyjnego, bibliotek, izolacji od innych komponentów oprogramowania itp.

• Większe koszty operacyjne: Systemy rozproszone są bardziej złożone, wymagają więcej zadań związanych z zarządzaniem wykonywanych przez operatorów.

• Mniejsza wydajność energetyczna: marnotrawstwo energii na dedykowanych serwerach może zwiększyć zużycie energii przez hosty i systemy chłodzenia.

• Wymagana większa ilość miejsca: nawet zgrupowane w szafach lub serwerach kasetowych wymagają więcej miejsca niż integracja komputerów mainframe.

• Potencjalnie mniejsza wydajność we/wy: przechowywanie danych jest scentralizowane w komputerach mainframe, ale systemy rozproszone, które uzyskują dostęp przez sieć do scentralizowanej pamięci masowej, mają opóźnienia w dostępie do nich.

• Potencjalnie trudniej jest być odpornym na awarie: Ponieważ sieć i liczba węzłów usługowych wprowadzają więcej punktów awarii. O ile komponenty sprzętowe w komputerach mainframe również

są podatne na ataki, implementują one mechanizmy nadmiarowości, aby zminimalizować wpływ awarii.

• Brak możliwości współdzielenia zasobów w węzłach rozproszonych: wolne moce procesora w jednym węźle nie mogą być wykorzystywane przez inne węzły. W tym przypadku możliwość skonsolidowania liczby procesorów w jednym komputerze mainframe miała również istotny wpływ ekonomiczny pod względem oszczędności kosztów licencji oprogramowania warstwy pośredniej (ponieważ ceny większości licencji oprogramowania warstwy pośredniej rosną w zależności od liczby procesorów).

Na początku 2000 roku te wady sprawiły, że niektórzy menedżerowie IT ponownie rozważyli powrót do odnowionych architektur mainframe, które umożliwiły konsolidację zasobów sprzętowych przy jednoczesnej możliwości uruchamiania wielu instancji systemu operacyjnego. Architektury te wprowadziły technologie wirtualizacji (takie jak IBM z/Virtual Machine i z/Virtual Storage Extended) oraz partycje logiczne, które mogą obsługiwać system UNIX (głównie Linux) lub natywny system operacyjny mainframe (VMS, z/OS itp.). Nowoczesne komputery mainframe potrafią również dynamicznie rekonfigurować zasoby przypisane do maszyny wirtualnej lub partycji logicznej (procesory, pamięć, połączenia urządzeń itp.). W świecie systemów rozproszonych technologie klastrowania, wirtualizacji (VMware4 lub Xen5) i narzędzia do automatyzacji centrów danych (Tivoli6 firmy IBM lub OPSware HP) znacznie ułatwiają zarządzanie centrami danych, minimalizując złożoność i koszty operacyjne udostępniania systemów rozproszonych (alokacja sieci i zasoby obliczeniowe, instalacja systemu operacyjnego i komponentów oprogramowania itp.) i zarządzanie (łatki systemu operacyjnego, aktualizacje oprogramowania, monitorowanie, zasilanie itp.). Nowoczesne centra danych są zorganizowane w klastry. Klaster to grupa komputerów standardowych, ściśle współpracujących ze sobą, dzięki czemu pod wieloma względami tworzą jeden komputer. Komponenty klastra są często, choć nie zawsze, połączone za pośrednictwem szybkich sieci lokalnych. Klastry są zwykle wdrażane w celu poprawy wydajności (poprzez skalowanie dostępnej mocy obliczeniowej) i/lub dostępności w porównaniu z pojedynczym komputerem, przy czym zazwyczaj są one znacznie bardziej opłacalne niż pojedyncze komputery o porównywalnej szybkości lub dostępności. Jednak łączenie podstawowych zasobów (często obliczeniowych) w klaster często nie wystarczało dla wysokiego zapotrzebowania obliczeniowego wymaganego przez niektóre aplikacje i eksperymenty. Potrzebne było dalsze skalowanie, aby zaabsorbować rosnący popyt. Konieczność agregowania rzadkich i oddzielnie zarządzanych zasobów obliczeniowych dała początek koncepcji Organizacji Wirtualnych. Są to oddzielne domeny administracyjne, które ustalają sposoby korzystania z niewykorzystanych zasobów należących do innych współpracujących organizacji tak, jakby były zlokalizowane lokalnie. Rzeczywiście, wirtualizacja umożliwia konsolidację różnych maszyn wirtualnych na tym samym fizycznym hoście, zmniejszając straty mocy obliczeniowej i energii. Rzeczywiście, z punktu widzenia architektury komputerów, ewolucja komputerów nie bardzo wyraźnie pokazuje różnicę między serwerami dedykowanymi a komputerami typu mainframe, ponieważ komputery typu mainframe mogą być uważane za najwyższy model w rodzinie komputerów. Następnie kluczową kwestią w strategii usług IT może być skalowanie w poziomie z tanimi serwerami (w postaci klastrów lub małych serwerów) lub w pionie z „dużymi" serwerami pamięci współdzielonej. Jak zwykle nie ma srebrnej kuli i decyzja może zależeć od aplikacji. W każdym razie dzięki wprowadzeniu wirtualizacji podstawowa fizyczna warstwa sprzętowa jest przezroczysta dla węzłów usługowych, które potencjalnie mogą skalować się w pionie do maksymalnej pojemności maszyn fizycznych i poziomo do maksymalnej pojemności centrum danych. Niektóre badania, zob. (Michael, Moreira, Shiloach i Wisniewski, 2007; Barroso i Hölzle, 2009), stwierdzają na przykład, że skalowanie poziome oferuje lepszy stosunek ceny do wydajności, chociaż przy wzroście złożoności zarządzania, w przypadku aplikacji webowych. Dodatkowo skalowanie poziome jest uważane przez niektórych autorów za jedyne rozwiązanie dla superkomputerów.

Komputery magazynowe

Firmy internetowe, takie jak Google, Amazon, Yahoo i dział usług online Microsoftu, przekształciły swoje centra danych za pomocą komputerów w skali magazynu (WSC), które różnią się od tradycyjnych centrów danych:

• Centra danych należą do jednej firmy.

• Stosowanie stosunkowo jednorodnych platform sprzętowych i programowych.

• Duży klaster traktowany jako pojedyncza jednostka obliczeniowa, a nie tylko zestaw przewodowych pojedynczych serwerów.

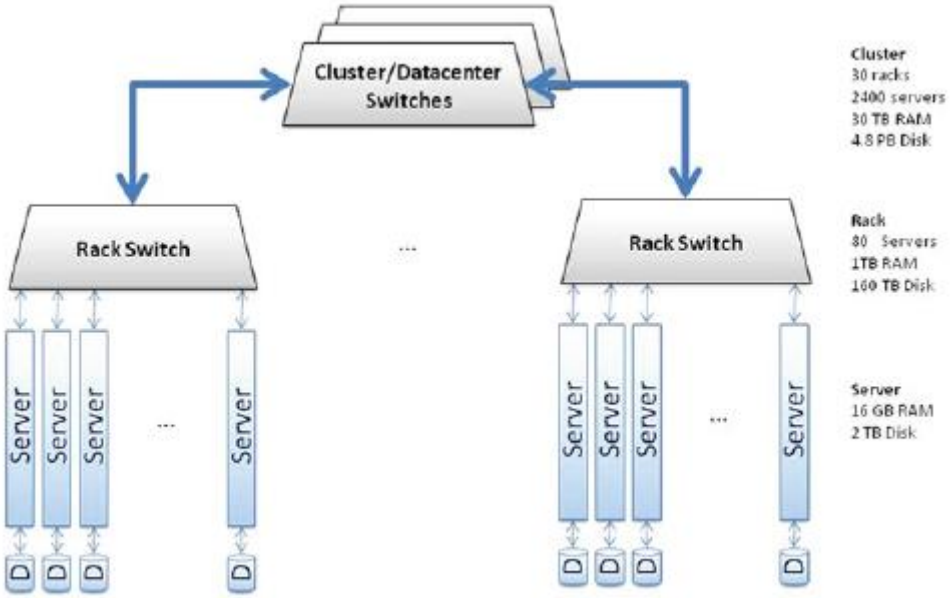
• Często większość platformy oprogramowania (aplikacje, oprogramowanie pośredniczące i oprogramowanie systemowe) jest budowana we własnym zakresie i dostosowywana do świadczonych przez nich usług (wyszukiwarki, hurtownie mediów, e-commerce itp.) zamiast korzystania z oprogramowania stron trzecich (standard serwery aplikacji, oprogramowanie pośredniczące, system operacyjny itp.).

• Uruchamiają mniejszą liczbę bardzo dużych aplikacji.

• Wykorzystanie wspólnej warstwy zarządzania, która elastycznie kontroluje wdrażanie aplikacji wśród współdzielonych zasobów.

• Wysoka dostępność osiągnięta przy założeniu dużej liczby usterek komponentów o niewielkim lub zerowym wpływie na poziom usług.

Rysunek przedstawia typową architekturę systemów na skalę magazynową , której podstawowymi elementami są tanie serwery 1U lub serwery kasetowe montowane w stelażu.



Serwery są połączone ze sobą przełącznikami Ethernet 1–10 Gb/s na poziomie szafy z połączeniami uplink do jednego lub większej liczby przełączników Ethernet na poziomie klastra lub centrum danych. Dyski mogą być zarządzane przez Network Attached Storage (NAS) podłączony bezpośrednio do przełączników klastra lub być podłączone do każdego indywidualnego serwera i zarządzane przez

globalny rozproszony system plików, taki jak system plików Google (GFS). Jak pokazano powyżej, narzędzia do zarządzania zasobami są kluczem do kontrolowania dynamicznego udostępniania zasobów i wdrażania aplikacji, a co za tym idzie, do stopniowego skalowania. Ponownie, do skalowania systemów wykorzystywane są również określone architektury i technologie aplikacji rozproszonych: rozproszone systemy plików, algorytmy paralelizacji, przekazywanie komunikatów itp. Centra danych można replikować w różnych lokalizacjach geograficznych, aby zmniejszyć opóźnienia użytkowników i poprawić wydajność usług.

Siatki i chmury

W przeciwieństwie do systemów na skalę magazynową pojawiły się technologie Grid i Cloud, które umożliwiają współdzielenie zasobów między organizacjami. Nazywane również infrastrukturami zorientowanymi na usługi, ich cel jest bardziej „ogólny", ponieważ muszą obsługiwać wiele różnych aplikacji z różnych domen i typów organizacji (badania, zarządzanie lub przedsiębiorstwa). Zresztą większość funkcji, które powinny zapewniać, jest wspólnych, a funkcje „publiczne" mogą zostać przejęte przez infrastrukturę „prywatną" i odwrotnie. Grid jest jedną z takich technologii rozszerzających skalę systemów obliczeniowych poprzez agregację zasobów towarowych należących do różnych domen administracyjnych w jedną lub więcej organizacji wirtualnych. Bardziej formalnie, Grid definiuje się jako „system koordynujący zasoby, które nie podlegają scentralizowanej kontroli, wykorzystujący standardowe, otwarte protokoły i interfejsy ogólnego przeznaczenia w celu zapewnienia nietrywialnych jakości usług" (Foster, 2002). Nowsze definicje podkreślają zdolność łączenia zasobów z różnych organizacji dla wspólnego celu. W Kurdi, Li i Al-Raweshidy oraz Stockinger (2007) chodzi o koordynację zasobów z różnych dziedzin oraz o to, jak tymi zasobami należy zarządzać. Oprócz agregowania większej liczby organizacji w jedną wirtualną organizację, dana usługa była trudna do skalowania, a Grid, zgodnie z tradycyjnym pojmowaniem, nie oferował żadnych mechanizmów pomagających deweloperom Grid Service w skalowaniu ich systemów zgodnie ze zmianami popytu. Pod tym względem Grid nie różnił się od poprzednich systemów informatycznych (IT): administrator powinien wykrywać przeciążenia usług i ręcznie skalować system w oparciu o wskaźniki wydajności istotne dla danej usługi. Ponadto liczba węzłów, z których składa się organizacja wirtualna, może być znacznie niższa od liczby potrzebnej do wykonania zamierzonego zadania. Niedawno na scenie pojawił się inny paradygmat, chmura, który ma pomóc w zwiększeniu skalowalności zapewnianej użytkownikowi końcowemu. Jednak różnice nie są jasne, być może dlatego, że chmury i sieci mają podobne wizje: zmniejszenie kosztów obliczeniowych oraz zwiększenie elastyczności i niezawodności przy użyciu sprzętu obsługiwanego przez strony trzecie (Vaquero i in., 2009). Na przykład sieci Grid zwiększają sprawiedliwe współdzielenie zasobów między organizacjami, podczas gdy chmury zapewniają wymagane zasoby na żądanie, sprawiając wrażenie pojedynczego dedykowanego zasobu. W związku z tym nie ma faktycznego współdzielenia zasobów ze względu na izolację zapewnianą przez wirtualizację. Niemniej jednak technologie wirtualizacji są również wykorzystywane do pomocy w skalowaniu Gridów na poziomie pionowym (np. dodawanie większej liczby zasobów do maszyny wirtualnej). Inna ważna różnica między tradycyjnymi Gridami a chmurą dotyczy zastosowanego modelu programowania. Na przykład użytkownik chmury może wdrażać aplikacje oparte na Enterprise Java Beans, tak jak może zamiast tego wdrożyć zestaw usług Grid. Chmura będzie traktować ich obu jednakowo. Jednak z definicji Gridy akceptują tylko aplikacje „zgridyzowane" (Vaquero i in., 2009; Stockinger, 2007), narzucając w ten sposób programistom twarde wymagania. Chociaż wirtualne organizacje dzielą koszty sprzętu (i zarządzania nim), nadal są one wyższe niż „wynajmowanie" wymaganej pojemności dokładnie wtedy, gdy jest ona potrzebna. Wskazuje to, że model provisioningu jest bardzo ważnym elementem, jeśli chodzi o określenie potencjalnej skalowalności naszego systemu informatycznego. W siatce, po lewej stronie Rys. 15.3 poniżej, żądanie zadania (1) zostało skojarzone przez brokera z niektórymi dostępnymi zasobami znalezionymi w serwisie informacyjnym (2), zasoby

są zarezerwowane w celu zagwarantowania odpowiedniego wykonania (3) i uporządkowane w systemie przepływu pracy (4) kontrolującym skoordynowane działanie przydzielonych zasobów (5). W chmurze infrastruktury jako usługi użytkownik jest odpowiedzialny za dostarczenie pakietu oprogramowania (1), a menedżer chmury znajduje dostępne zasoby (2,3) do hostowania maszyn wirtualnych zawierających stos oprogramowania (4). Reprezentują one dwie zupełnie różne filozofie dostarczania i zarządzania. Paradygmat przetwarzania w chmurze przenosi lokalizację zasobów obliczeniowych do sieci, aby obniżyć koszty związane z zarządzaniem zasobami sprzętowymi i programowymi. Udostępnianie zasobów na żądanie i skalowalność to niektóre z podstawowych cech chmury. Chmura oferuje wiele typowych punktów skalowania, których aplikacja może potrzebować, w tym serwery, pamięć masową i usługi sieciowe leżące u podstaw zmiany rozmiaru zasobów (w ramach infrastruktury jako usługi, IaaS, chmury) lub zaawansowane platformy programistyczne i konserwacyjne w celu skrócenia czasu świadczenia usługi na rynek (w Platform and Software as a Service, PaaS/SaaS, Clouds). Tak więc chmurę można zapewne zdefiniować nie jako postęp technologiczny, ale jako model dynamicznego świadczenia usług, gdzie usługa to wszystko, co może być oferowane jako usługa sieciowa . Charakter chmury obliczeniowej na żądanie w połączeniu z wyżej wspomnianym modelem płatności za rzeczywiste wykorzystanie oznacza, że wraz ze wzrostem zapotrzebowania na aplikacje mogą również rosnąć zasoby, których używasz do obsługi tego zapotrzebowania. W takiej sytuacji system w końcu osiąga równowagę, a przydzielona pojemność jest równa zapotrzebowaniu, o ile aplikacja została poprawnie zaprojektowana, a jej architektura jest podatna na odpowiednie skalowanie. Idealnie, aplikacje we wdrożeniach IaaS Cloud powinny działać zgodnie z wysokopoziomowymi celami i nie przedstawiać administratorom konkretnych szczegółów implementacji. Istniejące strategie wymagają od programistów przepisania swoich aplikacji, aby wykorzystać wykorzystanie zasobów na żądanie, blokując w ten sposób aplikacje w określonej infrastrukturze chmury. Niektóre podejścia strukturyzują serwery w hierarchiczne drzewo, aby osiągnąć skalowalność bez znaczącej restrukturyzacji bazy kod. Profile służą również do gromadzenia wiedzy ekspertów na temat skalowania różnych typów aplikacji. Podejście oparte na profilach automatyzuje wdrażanie i skalowanie aplikacji w chmurze bez wiązania się z określoną infrastrukturą chmury . Można zastosować podobne metody, które analizują wzorce komunikacji między operacjami usług i przyporządkowanie zaangażowanych usług do dostępnych serwerów, optymalizując w ten sposób strategię alokacji w celu poprawy skalowalności usług złożonych. Potrzeba powyższych strategii i przepisywania kodu wyraźnie wskazuje na trudności w skalowaniu aplikacji w chmurze. Chociaż podjęto pewne niezwykłe próby, które mają na celu dodanie możliwości automatycznego skalowania do systemów opartych na usługach , są one często trudne do opracowania, zbyt zależne od specyficzne zastosowanie i trudno je uogólnić, aby były oferowane jako usługa ogólnego przeznaczenia. W związku z tym obecne komercyjne systemy Grid i Cloud opierają się również na know-how użytkownika w zakresie „maksymalnej” pojemności i budują do tej pojemności. Oznacza to, że system jest zwykle w jednym z dwóch trybów: niedokupienie lub przekupienie. W przeciwieństwie do sieci Grid i poprzednich metod udostępniania systemów informatycznych, chmura umożliwia dostawcom usług proste zwiększanie pojemności w razie potrzeby, zazwyczaj z czasem realizacji wynoszącym kilka minut. Model płatności za rzeczywiste wykorzystanie pozwala płacić tylko za to, co jest faktycznie udostępniane. Pomimo elastycznych modeli dostarczania i rozliczania, stopień automatyzacji i integracji z podstawowymi systemami monitorowania oferowanymi przez większość istniejących systemów Cloud ma być dalej rozwijany. Kluczem do skutecznego skalowania na żądanie są dokładne metryki wykorzystania. Na pierwszym planie funkcje Amazon Cloud Watch i Auto-scale11 umożliwiają pewną integrację podstawowego systemu monitorowania (zapewniając wskaźniki infrastruktury, takie jak wykorzystanie procesora) oraz usługę reagowania na warunki zdefiniowane przez użytkownika. Ponadto RightScale umożliwia zautomatyzowanie niektórych działań wyzwalających w oparciu o metryki infrastruktury lub skrypty zdefiniowane przez użytkownika umieszczone na wdrożonych

serwerach. Jednak stopień dostosowania reguł skalowania przez użytkowników zależy od tego, a metryki na poziomie aplikacji są trudne do uwzględnienia, co skutkuje niską automatyzacją reguł lub metryk wysokiego poziomu (np. zgłoszeń serwisowych w okresie), które są bliższe sposób myślenia użytkownika chmury. Poza tym, dodawanie większej liczby maszyn na żądanie, podstawowe technologie wirtualizacji nieodłącznie związane z chmurą, umożliwiają uwzględnienie skalowania pionowego (więcej zasobów można dodać w locie do maszyny wirtualnej). Niestety, ta bardzo pożądana funkcja nie jest jeszcze obsługiwana przez większość systemów operacyjnych (wymagają stałego rozmiaru procesora i pamięci w czasie rozruchu). Podsumowując, chociaż chmury IaaS poszły o krok dalej w skalowalności systemu, definiowanie automatycznych działań skalowalności na podstawie niestandardowych metryk usług nie jest na dzień dzisiejszy obsługiwane. W rzeczywistości żadna platforma Cloud nie obsługuje konfiguracji pewnych reguł biznesowych, takich jak na przykład limity maksymalnych wydatków, które SP jest gotów zapłacić, aby nie zbankrutować na przykład z powodu ataków Economic Denial of Sustainability (EDoS). Chociaż chmury i sieci IaaS mogą różnić się potencjałem skalowalności ,chmury PaaS/SaaS wciąż muszą poczynić ogromne postępy w zakresie pomocy w zwiększeniu wymaganych poziomów automatyzacji i abstrakcji. Skalowalność just-in-time nie jest osiągana przez proste wdrażanie aplikacji w chmurze. Skalowanie SaaS to nie tylko posiadanie skalowalnego podstawowego (wirtualnego) sprzętu, ale także pisanie skalowalnych aplikacji. Cenne zasady zostały dostarczone przez platformy PaaS, takie jak Google App Engine. Obejmują one minimalizację pracy, stronicowanie przez duże zestawy danych, unikanie rywalizacji o magazyn danych, liczniki fragmentów i efektywną pamięć podręczną. Techniki te wskazują, że na dzień dzisiejszy tradycyjne techniki pomagające w skalowaniu aplikacji muszą pochodzić od programistów tworzących punkty skalowania.

Potrzebne są bardziej wyrafinowane środki, aby klienci nie musieli stać się ekspertami w zarządzaniu wątkami lub programami zbierania śmieci. Sama chmura powinna zapewniać takie udogodnienia dla programistów jako usługa (PaaS). Jedną z dużych zalet chmury jest to, że można jej użyć do podzielenia programu, tak aby różne instrukcje mogły być przetwarzane w tym samym czasie. Ale trudno jest napisać kod potrzebny do tego w większości języków programowania. Na przykład C i inne języki programowania oferowały interfejsy API MPI do wykonywania zrównoleglania instrukcji w klastrze obliczeniowym. Jednak tradycyjne środowiska programistyczne dostarczają nieodpowiednich narzędzi, ponieważ obciążają programistów, którzy powinni działać na zbyt niskim poziomie abstrakcji. Jednym z takich przedsięwzięć jest projekt BOOM (Berkeley Orders Of Magnitude), którego celem jest zbudowanie łatwo skalowalnej dystrybucji z mniejszą ilością kodu, w tym GFS, Map/Reduce i Chubby, a następnie umożliwienie rekonfiguracji i ponownej kompozycji tych komponentów w celu umożliwienia nowych systemów rozproszonych. łatwe do zbudowania przy niskich kosztach (Boom09, 2009). Brak takich mechanizmów skutkuje konkretnymi rozwiązaniami, które trudno uogólniać i oferować jako usługę. Bardziej ogólne podejścia dotyczą wiedzy o domenie aplikacji w celu zwiększenia skalowalności aplikacji poprzez zminimalizowanie zmian, które należy wprowadzić w kodzie aplikacji. Zostało to zastosowane na przykład w sieciach społecznościowych online dzięki wykorzystaniu struktury wykresu. Grupy są rozdzielone na różnych serwerach, a węzły należące do kilku grup są replikowane na wszystkich serwerach grupy

Skalowalność aplikacji

Powyższy przykład (Pujol et al., 2009) wyraźnie wskazuje, że nie wszystkie aplikacje są dobrze przystosowane do skalowania w chmurze. Chociaż jest to idealne środowisko dla aplikacji internetowych, to aplikacje transakcyjne nie mogą być „zachmurzone” w tak łatwy sposób. Aplikacje internetowe mogą być skalowane w poziomie iw pionie oraz rozłożone na kilka centrów danych bez zapór sieciowych (lub innych problemów związanych z siecią). Aplikacje internetowe są zwykle

bezstanowe, co oznacza, że migracja usług z jednej lokalizacji do innej nie oznacza żadnych uchybień w wydajności aplikacji. Ponadto w miarę dodawania nowych replik (skalowanie poziome) systemy równoważenia obciążenia mogą równomiernie przekierowywać żądania do dowolnej dostępnej repliki. Jednak baz danych nie można tak łatwo przenieść do chmury. Nie opierają się na protokołach gotowych do korzystania z Internetu (takich jak HTTP), więc problemy można znaleźć w wysoce rozproszonym i wielodostępnym środowisku, takim jak chmura. Ponadto są one z natury stanowe, a wycofywanie i zatwierdzanie są funkcjami niezbędnymi do odpowiedniego zachowania usługi. Ta ostatnia przesłanka oznacza, że usługi nie mogą być migrowane ani nigdzie zlokalizowane. Ograniczenia prawne ograniczają również migrację i faktyczną lokalizację niektórych bardzo wrażliwych danych (chociaż te nietechniczne implikacje wykraczają poza zakres tego rozdziału). Replikacja bazy danych jest zwykle wykonywana przez doświadczonych administratorów i bardzo często zależy od konkretnego modelu danych. Strategia replikacji wpływa na równoważenie obciążenia, co dodatkowo komplikuje zapewnienie aplikacjom transakcyjnym automatycznej skalowalności. Nowe transakcyjne aplikacje SaaS powinny opierać się na kilku podstawowych koncepcjach programistycznych używanych przez aplikacje internetowe w celu osiągnięcia wysokiej wydajności lub wysokiej dostępności we wdrożeniach na dużą skalę (Barroso & Hölzle, 2009), nie próbując emulować tradycyjnych architektur transakcyjnych:

• Replikacja danych (dzięki temu aktualizacje są bardziej złożone)

• Sharding (partycjonowanie) zbioru danych na mniejsze fragmenty i dystrybucja ich pomiędzy dużą liczbą serwerów.

• Dynamiczne równoważenie obciążenia poprzez promowanie zasad shardingu w celu wyrównania obciążenia na węzeł.

• Kontrola stanu i liczniki czasu nadzoru: bardzo ważne jest, aby wiedzieć, czy serwer jest zbyt wolny lub nieosiągalny, aby jak najszybciej podjąć działania. Należy stosować limity czasu dla żądań zdalnych i techniki pulsu.

• Kontrole integralności w celu uniknięcia uszkodzenia danych

• Kompresja specyficzna dla aplikacji

• Spójność ostateczna: systemy wielkoskalowe osłabiają spójność danych w ograniczonych okresach, które ostatecznie powróci do stabilnego, spójnego stanu.

Automatyzacja skalowalności

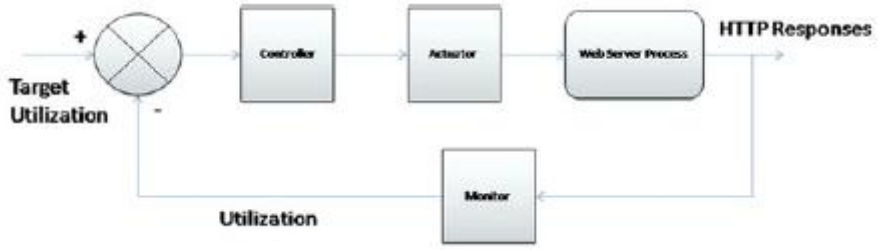
Automatyczna skalowalność na poziomie aplikacji może być realizowana na kilka sposobów. Dwa najważniejsze z nich to:

1. Użytkownicy dostarczają zestaw reguł w dobrze zdefiniowanym języku (np. patrz Galán i in. (2009)), który zasila „kontrolera aplikacji" działającego w imieniu użytkownika i odpowiedzialnego za egzekwowanie określonych reguł skalowalności.

2. Zaprojektuj i zaimplementuj algorytmy lub metody statystyczne specyficzne dla aplikacji, aby kontroler wiedział, kiedy aplikacja powinna się skalować, bez konieczności uciekania się do bezpośrednich działań użytkowników w systemie eksperckim.

Jeśli chodzi o pierwsze podejście powyżej, bardzo istotny przykład zostanie podany poniżej (patrz sekcja 15.3.2). Wystarczy powiedzieć, że systemy oparte na regułach dostarczają znaczących opisów odpowiednich warunków do skalowania, a techniki eksploracji danych można wykorzystać do

wyodrębnienia odpowiednich reguł i pomocy użytkownikom (dostawcom usług) w zdobyciu wiedzy na temat wydajności aplikacji i technik optymalizacji. Z drugiej strony, techniki algorytmiczne (mamy przez to na myśli również narzędzia statystyczne, sieci neuronowe lub tradycyjne techniki teorii sterowania) zapewniają pewien stopień kontroli w czasie rzeczywistym, której nie mogą jeszcze osiągnąć systemy oparte na regułach. Ta sekcja koncentruje się na niektórych funkcjach tych systemów. Rysunek 15.5 opisuje rozwiązanie do zarządzania QoS (Quality of Service) dla aplikacji internetowej opartej na teorii sterowania.



Rozszerzając te koncepcje na aplikację w chmurze IaaS, pętla sterowania, która stara się uniknąć przeciążenia systemu i spełnić indywidualne gwarancje czasu odpowiedzi i przepustowości, byłaby:

• Aplikacja serwisowa: aplikacja serwisowa, która ma być wykonywana na jednej lub kilku maszynach wirtualnych w chmurze.

• Monitor: zapewnia informację zwrotną o wykorzystaniu zasobów na podstawie dostępnych miar, takich jak procesor, pamięć dyskowa i przepustowość sieci.

• Kontroler: biorąc pod uwagę różnicę między pożądaną jakością QoS a wykorzystaniem zasobów (mierzonym przez komponent monitora), kontroler musi zdecydować o działaniach naprawczych, aby osiągnąć docelową jakość QoS. Tak więc teoria sterowania oferuje techniki analityczne do zamykania pętli, takie jak modelowanie systemu jako funkcji wzmocnienia, która maksymalizuje wykorzystanie zasobów między dopuszczalnymi marginesami.

• Aktuator: tłumaczy abstrakcyjne dane wyjściowe kontrolera na konkretne działania podejmowane przez oprogramowanie pośredniczące w chmurze w celu skalowania w górę/w dół składników usługi w celu zmiany jej obciążenia.

W tym miejscu podkreślamy otwarte wyzwanie dla dynamicznej skalowalności w chmurze: znalezienie kontrolera, który mógłby modelować usługę wdrożoną w chmurze i sprawić, by system skalował się zgodnie z oczekiwaniami. W Rosu, Schwan, Yalamanchili i Jha (1997) autorzy zaproponowali wykorzystanie adaptacyjnej alokacji zasobów (ARA) w czasie rzeczywistym dla platform systemów wbudowanych. Mechanizmy ARA szybko dostosowują alokację zasobów (skalowalność pionową) do zmian w zmienności środowiska wykonawczego potrzeb aplikacji, ilekroć istnieje ryzyko niespełnienia ograniczeń czasowych, unikając „przewymiarowania" systemów czasu rzeczywistego w celu spełnienia najgorszego scenariusza. ARA modeluje aplikację jako zestaw połączonych ze sobą komponentów oprogramowania, których wykonanie jest sterowane przez strumienie zdarzeń:

• Model wykorzystania zasobów: opisuje oczekiwane potrzeby obliczeniowe i komunikacyjne aplikacji oraz ich odmiany w czasie wykonywania.

• Model adaptacyjny: konfiguracje akceptowalne pod względem oczekiwanych potrzeb w zakresie zasobów i kosztów ogólnych konfiguracji specyficznych dla aplikacji.

Model ten można wygenerować za pomocą statycznych i dynamicznych narzędzi profilowania, które analizują odpowiednio kod źródłowy i środowisko uruchomieniowe aplikacji pod różnymi

obciążeniami. Następnie kontroler ARA wykrywa ryzyko niespełnienia docelowej wydajności, dzięki czemu oblicza akceptowalną konfigurację i bardziej odpowiednią alokację zasobów. Szacunki zapotrzebowania na zasoby są dokonywane w oparciu o charakterystykę węzła (współczynnik szybkości procesora, szybkość łączy komunikacyjnych, obciążenie komunikacyjne) oraz statyczną aplikację (poziom równoległości, czas wykonania, liczbę wymienianych komunikatów i współczynnik szybkości procesora) i dynamiczną (współczynnik wykonania, wewnątrzkomponentowy czynniki wymiany wiadomości) modele wykorzystania zasobów. Podobne techniki czasu rzeczywistego oparte na matematycznym profilowaniu aplikacji i negatywnych pętlach sprzężenia zwrotnego, takie jak ARA, mogą być dostosowane do problemu automatyzacji skalowalności chmury i stanowić przyszłe wyzwania badawcze. Rzeczywiście, przewidywanie szeregów czasowych jest bardzo złożonym procesem i bardzo specyficznym dla określonej aplikacji domeny (lub nawet bardzo specyficzne aplikacje).

Skalowalne architektury

Ogólne architektury chmury do skalowania

Zaproponowano kilka architektur w celu ustrukturyzowania chmury i jej aplikacji. Buyya i in. zaproponował architekturę umożliwiającą QoS dla chmury, która przedstawiała księgowość, ustalanie cen, kontrolę dostępu i monitorowanie jako kluczowe elementy umożliwiające jej realizację. W oparciu o ten model, pełniejszy model architektoniczny został niedawno zaproponowany przez Brandic, Music, Leitner i Dustdar. Chociaż te architektury były zbyt zorientowane na QoS (kluczowy element dla chmury, aby stać się prawdziwą opcją dla dużych przedsiębiorstw), zidentyfikowano kilka istotnych wspólnych elementów. W oparciu o te wspólne elementy zaproponowano zintegrowaną architekturę stosu przetwarzania w chmurze, która ma służyć jako punkt odniesienia. Na podstawie tego uogólniamy go dalej, aby dać jasny przegląd. Pomimo powszechnej akceptacji, architektura ta jest zbyt ogólna, aby zilustrować najważniejsze kwestie do tej pory poruszone. W tej sekcji zajmowaliśmy się problemami skalowalności w różnych warstwach chmury. Tutaj podsumowujemy główne mechanizmy skalowalności w chmurze na dzień dzisiejszy. Natomiast skalowalność i podstawowa automatyzacja mechanizmy niskopoziomowe zostały już zaimplementowane dla chmur IaaS (a wiele innych ma być rozwijanych w kierunku pełnej automatyzacji i odpowiedniego poziomu abstrakcji). Na poziomie PaaS wciąż ma nastąpić okres „kambryjski", aby zwiększyć liczbę technik pomagających skalować aplikacje w chmurze w sposób zgodny wstecznie. Oczekuje się, że te techniki zostaną zaimplementowane przez samego programistę, a nie pomogą w tworzeniu naprawdę zaawansowanych usług skalowania w oparciu o usługi oferowane przez platformę. Ta zmiana przeniesie obecne wysiłki w zakresie skalowania SaaS z programistów na podstawową platformę.

Przykład paradygmatyczny: skalowalność zbiornika

Zasoby i usługi Wirtualizacja bez barier to projekt współfinansowany przez Unię Europejską z 7PR, który umożliwia wdrażanie i zarządzanie złożonymi usługami na masową skalę w różnych domenach administracyjnych. RESERVOIR ma strukturę warstwową, która zasadniczo może być zawarta w powyższej warstwie IaaS (zarządzanie zasobami zwirtualizowanymi i fizycznymi oraz klastrem tych zasobów). Jedną z zasadniczych innowacji w tym przedsięwzięciu w odniesieniu do skalowalności jest zdefiniowanie technik i technologii dla federacji kilku niezależnie działających obiektów RESERVOIR. Federacja pozwala danemu dostawcy chmury na outsourcing niektórych usług i „wydzierżawienie" zasobów zewnętrznych w sposób płynny. Jednak ta strategia skalowalności jest nieco podobna do tej stosowanej przez Clouds w porównaniu do Grids. Wymagane są dodatkowe środki, aby umożliwić odpowiednią architekturę zapewniającą skalowalność usług sieciowych. Aby przezwyciężyć obecne ograniczenia skalowalności IaaS, RESERVOIR proponuje nową warstwę abstrakcji bliższą cyklowi życia

usług, która pozwala na ich automatyczne wdrażanie i eskalację w zależności od stanu usługi (nie tylko od stanu infrastruktury). Te funkcje wykraczają poza czyste możliwości IaaS, ale pozostają kluczowe dla rzeczywistej skalowalności w różnych warstwach. Warstwa RESERVOIR Service Management (SM) automatycznie obsługuje cykl życia usługi i automatycznie skaluje usługi w zależności od „reguł" i metryk skalowania poziomu usług. Odbywa się to za pomocą abstrakcji bliższej tej zarządzanej przez programistę usług. W poniższym przykładzie reguła składa się z warunku (liczba zadań na węzeł executorów jest większa niż 50 i są uruchomione mniej niż 3 executory) oraz skojarzonej akcji, jeśli reguła jest spełniona (utwórz nową replikę). Jest to określone w standardowy sposób

<rsrvr:Rule>

<rsrvr:RuleName>rule2</rsrvr:RuleName>

<rsrvr:RuleType>AgentMeasureEvent</rsrvr:RuleType>

<rsrvr:Trigger checkingPeriod="5000ms"

condition="(@{kpis.QueueLength}/(@{components.VEEBigExecutor.

replicas.amount}*3 + @{components.VEEExecutor.replicas.

amount} +1)> 50) && (@{components.VEEExecutor.

replicas.amount} < 3)"/>

<rsrvr:Action run="createReplica(components.VEEExecutor)"/>

</rsrvr:Rule>

Pomimo tych znaczących postępów w zakresie chmur IaaS, RESERVOIR nie zajmuje się skalowalnością PaaS ani SaaS, pozwalając na rozwiązanie ważnej części problemu przez przyszłe prace badawcze.

Wnioski i kierunki na przyszłość

Skalowalność podążała zakręconą linią stosowaną do komputerów mainframe, systemów rozproszonych, z powrotem do komputerów mainframe i z powrotem do „scentralizowanej" chmury, która jest rozproszona i heterogeniczna, ale postrzegana jako pojedyncza jednostka przez urządzenia brzegowe uzyskujące dostęp do chmury za pośrednictwem standardowych interfejsów w celu wykonania usługi. W związku z tym możliwości skalowania oscylowały między skalowalnością poziomą i pionową, podążając za dominującym trendem projektowania i wdrażania systemów. W tym rozdziale pokrótce omówiono główne cechy skalowalności oferowane przez niektóre z najbardziej znaczących systemów scentralizowanych i rozproszonych, które pojawiły się do dnia dzisiejszego. Tutaj pokazaliśmy niektóre z najbardziej znanych przykładów skalowalności z obsługą chmury w różnych warstwach chmury. Należy zauważyć, że skalowalność jest oferowana w sposób przejrzysty dla użytkownika końcowego (dostawcy usług lub konsumenta usług). Maszyny wirtualne w chmurach IaaS można skalować w poziomie (poprzez dodanie większej liczby replik usług do danej usługi) lub w pionie (poprzez przypisanie większej liczby zasobów do pojedynczej maszyny wirtualnej). Same chmury IaaS można skalować w pionie (dodając więcej klastrów lub zasobów sieciowych) lub uzyskując funkcje federacji z innymi zewnętrznie zarządzanymi centrami danych (federacja chmury). Użytkownicy są całkowicie nieświadomi tych funkcji skalowania i dostarczani z iluzją nieskończonych zasobów. Skalowalność IaaS jest jednak nadal zbyt zorientowana na poziom usług, co oznacza, że decyzje dotyczące skalowania są podejmowane na podstawie czystych metryk infrastrukturalnych. W związku z tym nadal wymagane jest zaangażowanie użytkownika w zarządzanie usługami (nie dotyczy to zarządzania maszyną wirtualną, w której osiągnięto odpowiedni stopień automatyzacji). Pełna

automatyzacja i zastosowanie reguł skalowalności (lub modeli opartych na profilach obciążenia) w celu kontrolowania usług w sposób holistyczny są przyznawane na potrzeby przyszłych prac nad chmurami IaaS. Te zaawansowane funkcje automatyzacji zarządzania na wysokim poziomie są zbliżone do wyżej wymienionych funkcji PaaS. Zajmują się jednak tylko etapami wdrażania i cyklu życia usługi w czasie wykonywania. Więcej elementów pomagających skrócić czas wprowadzania usług na rynek i zapewniających dalsze wsparcie w zakresie projektowania aplikacji, rozwoju, debugowania, wersjonowania, aktualizacji itp. są nadal bardzo potrzebne. Na przykład, niektóre testy porównawcze specyficzne dla środowisk Cloud są już w toku.. Opracowane zostaną również konkretne benchmarki dla potencjału skalowania aplikacji w chmurze. Ogólnie rzecz biorąc, aplikacje skalujące w chmurze będą musiały stawić czoła „staromodnym wyzwaniom", wykrywając równoległość kodu (która może być oferowana jako usługa w chmurze PaaS), dystrybuując komponenty aplikacji w klastrach i operując usługami w rdzeniach (dla architektur wielordzeniowych) pozostaną przedmiotem masowych badań w przyszłych chmurach wspierających faktycznie skalowalne aplikacje w chmurze. Te udogodnienia w całym cyklu życia usług, w tym skalowalność usług, pomogą dostarczać zaawansowane usługi w krótszym czasie i minimalizować obciążenia związane z zarządzaniem. Niektóre inne problemy pozostają jednak do rozwiązania. Aby niezawodnie skalować usługi w chmurze dla milionów deweloperów usług i miliardów użytkowników końcowych, infrastruktura przetwarzania w chmurze i centrów danych nowej generacji będzie musiała przejść ewolucję podobną do tej, która doprowadziła do stworzenia skalowalnych sieci telekomunikacyjnych.