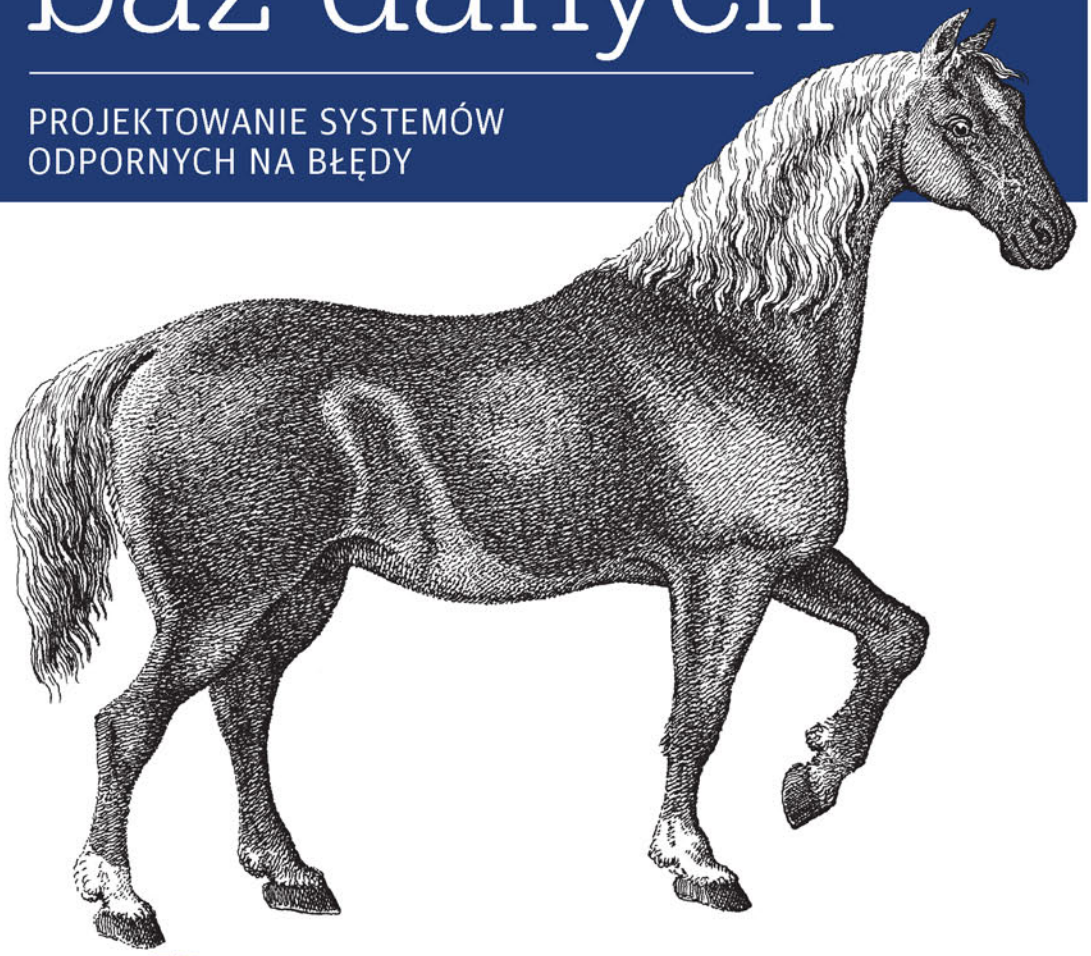
O'REILLY®

Inżynieria niezawodnych baz danych

PROJEKTOWANIE SYSTEMÓW ODPORNYCH NA BŁĘDY



Helion

Laine Campbell, Charity Majors

Tytuł oryginału: Database Reliability Engineering: Designing and Operating Resilient Database Systems

Tłumaczenie: Tomasz Walczak

ISBN: 978-83-283-4426-6

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

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

Printed in Poland.

Spis treści

ROZDZIAŁ 3.

Zarządzanie ryzykiem

Eksploatacja obejmuje zestaw obietnic i pracę potrzebną do ich spełnienia. W rozdziale 2. opisano, jak tworzyć takie obietnice, monitorować je i informować o ich realizacji. Zarządzanie ryzykiem polega na identyfikowaniu, ocenie i określaniu priorytetów związanych z niewiadomymi, które mogą skutkować naruszeniem tych obietnic. Ważne jest tu także wykorzystanie zasobów (technologii, narzędzi, ludzi i procesów) do monitorowania tych niewiadomych, łagodzenia ich skutków i ograniczania prawdopodobieństwa ich wystąpienia.

Nie jest to nauka ścisła. Celem nie jest wyeliminowanie wszystkich zagrożeń — to nierealny cel, który prowadzi do marnowania zasobów. Zadanie polega na wbudowaniu oceny i zmniejszaniu wpływu zagrożeń we wszystkie procesy oraz na stopniowym ograniczaniu skutków tych zagrożeń za pomocą technik łagodzenia problemów i zapobiegania im. Ten proces należy wykonywać stale na podstawie danych z obserwacji incydentów, wprowadzania nowych komponentów w architekturze i wzrostu lub spadku wpływu skutków wraz ze zmianami w organizacji. Przebieg tego procesu można podzielić na siedem etapów.

- Identyfikowanie możliwych zagrożeń i niebezpieczeństw, które generują ryzyko operacyjne dotyczące usługi.
- Ocena każdego ryzyka z uwzględnieniem prawdopodobieństwa i skutków.
- Kategoryzowanie prawdopodobieństwa i skutków zagrożeń.
- Określanie mechanizmów kontrolnych służących do łagodzenia skutków lub zmniejszania prawdopodobieństwa zagrożeń.
- Ustalanie priorytetów, czyli tego, którymi zagrożeniami trzeba się zająć w pierwszej kolejności.
- Implementowanie mechanizmów kontrolnych i monitorowanie efektywności.
- Powtarzanie procesu.

Powtarzając ten proces, stosujesz podejście *Kaizen*, czyli ciągłe usprawnianie. Nigdzie nie jest ono tak ważne jak przy ocenie ryzyka, gdzie konieczne jest stopniowe modyfikowanie strategii.

Kwestie związane z ryzykiem

Jest wiele zmiennych, które mogą wpływać na jakość procesu oceny ryzyka (*http://www.au.af.mil/au/awc/awcgate/usmc/orm.pdf*). Można je podzielić na następujące kategorie:

- nieznane czynniki i złożoność,
- dostępność zasobów,
- czynniki ludzkie,
- czynniki grupowe.

Trzeba uwzględnić każdą z tych kategorii, aby zdefiniować realistyczny proces dla zespołu. Dlatego w kolejnych punktach omawiamy pokrótce każdą z nich.

Nieznane czynniki i złożoność

Czynnikiem zwiększającym trudność procesu oceny ryzyka jest złożoność współczesnych systemów. Im bardziej złożona i skomplikowana dziedzina, tym trudniej ludziom przenieść ich wiedzę na sytuacje, z którymi wcześniej się nie zetknęli. Tendencja do nadmiernego upraszczania zagadnień, aby można było łatwo sobie z nimi poradzić, jest nazywana *błędem redukcji*. To, co sprawdza się na etapie początkowej nauki, nie działa w trakcie nabywania zaawansowanej wiedzy. Występuje wiele nieznanych zagrożeń, a liczne z nich pozostają poza naszą kontrolą. Oto kilka przykładów.

- Wpływ innych klientów w środowiskach z hostingiem, takich jak rozwiązania Amazonu lub Google'a.
- Wpływ dostawców narzędzi zintegrowanych z infrastrukturą.
- Przesyłanie kodu przez inżynierów oprogramowania.
- Działania marketingowe skutkujące skokami obciążenia roboczego.
- Usługi na wcześniejszych i dalszych etapach przetwarzania.
- Łatki, zmiany w repozytorium i inne stopniowo wprowadzane modyfikacje oprogramowania.

W trakcie oceny zagrożeń w takich środowiskach pomocne jest rozwiązywanie problemów z danej dziedziny. Zespół ds. eksploatacji musi wykorzystać swoje grupowe doświadczenia i stale rozwijać wiedzę, aby tworzyć dokładniejsze modele na potrzeby planowania. Zespoły muszą też pogodzić się z tym, że nie da się uwzględnić wszystkich możliwości. Dlatego trzeba planować z myślą o nieznanym, tworząc odporne systemy.

Dostępność zasobów

Jeśli pracowałeś kiedyś w dziale, gdzie brakowało zasobów, lub w niedoinwestowanym startupie, wiesz, że próby zdobycia zasobów na potrzeby bieżących, proaktywnych procesów bywają… no cóż, trudne (czytaj: to syzyfowa praca). Dlatego możliwe, że będziesz miał cztery godziny lub może tylko 30 minut miesięcznie na zajęcie się procesami zarządzania ryzykiem. Dlatego musisz zapewnić wartość. Cena Twojego czasu i zasobów, jakie wykorzystujesz do łagodzenia problemów, musi być niższa niż koszty niepodejmowania tych działań. Oznacza to, że musisz bezwzględnie ustalać priorytety na podstawie prawdopodobieństwa i wpływu zagrożeń oraz dostępnego czasu. Twórz odporne systemy i wyciągaj wnioski z zachodzących incydentów.

Czynnik ludzki

Gdy ludzie zaczynają wykonywać zadania, może wystąpić wiele problemów (*http://bit.ly/2zyoBmm*). Jesteśmy genialni, ale w naszym podręczniku użytkownika znajduje się wiele rzeczy zapisanych małą czcionką. Oto kilka czynników, które mogą zakłócać procesy.

Syndrom bierności

Wiele osób zajmujących się eksploatacją pracuje pod kierownictwem menedżera lub razem z ludźmi bojącymi się ryzyka. Tacy ludzie cechują się inercją i wybierają bierność, ponieważ uważają, że zmiany niosą ze sobą większe ryzyko niż bezczynność. Ważne jest, aby w obliczu nieznanego przeprowadzić obliczenia, zamiast uciekać w bezczynność.

Ignorowanie znanych zagrożeń

Doświadczeni inżynierowie często ignorują znane zagrożenia, a koncentrują się na bardziej „egzotycznych" i rzadkich zdarzeniach. Przykładowo ktoś przyzwyczajony do radzenia sobie z przepełniającymi się dyskami może w większym stopniu koncentrować się na zdarzeniach z poziomu centrum danych i nie planować w odpowiedni sposób mechanizmów kontrolnych związanych z przestrzenią dyskową.

Lęk

Lęk — w zależności od osoby — można uznać zarówno za pozytywny, jak i za negatywny stresor. Niektórzy ludzie najlepiej funkcjonują w wysoce stresujących środowiskach, gdzie gra toczy się o wysoką stawkę, i wtedy dużo wnoszą w planowanie, łagodzenie skutków i pracę produkcyjną. Z kolei osoby przejawiające reakcje lękowe nieraz z powodu swoich obaw ignorują najgorsze scenariusze. Może to prowadzić do braku przygotowania w obszarze istotnych, narażonych na duże ryzyko komponentów i systemów. Ważne, aby dostrzegać takie reakcje w zespole.

Nadmierny optymizm

Inną ludzką tendencją w kontekście oceny ryzyka jest nadmierny optymizm. Często myślimy o sobie i o innych osobach z zespołu w jak najlepszy sposób. Może to powodować, że oceniamy sytuację przez pryzmat idealnych warunków — brak zmęczenia, nie rozpraszają nas inne incydenty, dostępni są młodsi pracownicy. Dotyczy to nie tylko ludzi, ale także zdarzeń. Czy pomyślałeś sobie kiedyś: „Nie ma możliwości, aby trzy dyski zawiodły tego samego dnia", a później doświadczyłeś takiej sytuacji z powodu partii niesprawnych dysków?

Trzeba też uwzględnić czynniki fizyczne (takie jak zmęczenie) zwiększające ryzyko, a także utrudniające ręczne radzenie sobie z problemami („gaszenie pożarów"). Gdy analizujesz pracę ludzi (np. ręczne wprowadzanie zmian i badania) oraz nieodłączne od niej zagrożenia, musisz zakładać, że pracownicy odpowiedzialni za eksploatację zajmujący się poważnym problemem zostali właśnie obudzeni po ciężkim dniu pracy. Prawdopodobnie tak nie będzie, ale trzeba rozważyć taką ewentualność. Ponadto w trakcie projektowania mechanizmów kontrolnych na potrzeby łagodzenia lub eliminowania zagrożeń trzeba uwzględnić to, że osoba ręcznie rozwiązująca problem będzie bardzo zmęczona, a może nawet będzie musiała gasić wiele pożarów jednocześnie.

Zmęczenie wezwaniami

Zmęczenie wezwaniami (ang. *pager fatigue*; *http://bit.ly/2zyfqCv*) następuje, gdy niepotrzebne lub zbyt częste wezwania skutkują znużeniem i przytłoczeniem. Powinieneś to uwzględnić, gdy decydujesz, ile alarmów (wymagających ręcznej reakcji i interwencji) zostanie wbudowanych w procesy monitorowania. Zmęczenie wezwaniami często wynika z fałszywych alarmów (dotyczących sytuacji, które nie są rzeczywistymi problemami; nieraz powodem tego są źle dobrane wartości progowe) lub ze stosowania alarmów zamiast ostrzeżeń dla trendów, które mogą stać się niebezpieczne w bliskiej przyszłości.

Czynniki grupowe

Podobnie jak u ludzi pojawiają się „ślepe punkty", tak i grupy cechują się dynamiką, która zakłóca proces zarządzania ryzykiem. Oto niektóre czynniki, o których warto pamiętać.

Polaryzacja grupy

Polaryzacja grupy, którą można nazwać też przesunięciem punktu ryzyka, następuje, ponieważ grupy zwykle podejmują bardziej skrajne decyzje niż pojedynczy ich członkowie. To zjawisko często powoduje zmianę pierwotnych poglądów. Jeśli np. pojedyncze osoby początkowo były ostrożne, po osiągnięciu konsensusu stają się dużo bardziej odporne na ryzyko. Z kolei odporność na ryzyko może się przerodzić w jego unikanie. Poszczególne osoby często nie chcą okazać się najbardziej konserwatywne w środowisku grupy. Może to spowodować, że zespół będzie akceptował ryzyko w znacznie większym stopniu niż to wskazane.

Przekazywanie ryzyka

Grupy zwykle cechują się też większą akceptacją ryzyka, gdy mogą obarczyć nim inne zespoły. Jeśli np. opracowuję plany dla zespołu ds. eksploatacji, mogę podejmować większe ryzyko, gdy wiem, że mam w odwodzie zespół ds. baz danych. W rozwiązaniu tego problemu pomaga budowanie poczucia odpowiedzialności i praca w wielofunkcyjnych zespołach, które nie mogą obarczać ryzykiem innych.

Przekazywanie podejmowania decyzji

Przekazywanie podejmowania decyzji może zachodzić, gdy zespoły przeszacowują ryzyko, aby można było przenieść na innych odpowiedzialność za konkretne wybory. Jeśli np. bardzo ryzykowne zmiany wymagają zgody CTO, który bierze tym samym odpowiedzialność za nie, ludzie mają tendencję do zawyżania ryzyka, co pozwala im przekazać odpowiedzialność za decyzję w górę hierarchii. Ten problem można ograniczyć, tworząc bardziej autonomiczne zespoły, polegające na wiedzy eksperckiej i doświadczeniu jednostek oraz zespołów zamiast na hierarchicznych procesach zatwierdzania decyzji.

Co możemy zrobić?

Faktem jest to, że proces zarządzania ryzykiem może stać się nadmiernie obciążający. Nawet wtedy, gdy zaangażowano dużo zasobów, zespoły nie są w stanie uwzględnić wszystkich zagrożeń, które mogą wpływać na dostępność, wydajność, stabilność i bezpieczeństwo. Dlatego sensowne jest opracowanie iteracyjnego procesu usprawnianego wraz z czasem. Dążenie do zapewnienia odporności na zagrożenia zamiast eliminowania ich wszystkich pozwala też w inteligentny sposób podejmować ryzyko w celu zapewnienia innowacji i usprawnień.

Naszym zdaniem cel polegający na wyeliminowaniu wszystkich zagrożeń nie jest właściwy. Systemy bez stresorów zwykle nie są wzmacniane i usprawniane. Ostatecznie stają się wrażliwe na nieznane i niezaplanowane czynniki. Systemy, które regularnie doświadczają stresorów i dlatego zostały zaprojektowane z myślą o odporności, zwykle łatwiej radzą sobie z nieznanymi zagrożeniami.

Zdaniem niektórych w systemach należy „wykorzystywać budżet przestojów", jak jest to określane w Google'u, i ryzykować w sytuacjach, gdzie można uzyskać znaczne korzyści przy akceptowalnym poziomie ryzyka. W Google'u jeśli budżet wynosi 30 minut przestojów na kwartał i nie został wykorzystany, zespół jest gotów podejmować większe ryzyko, aby udostępnić nowe funkcje, usprawnienia i dodatki. Jest to doskonały sposób na wykorzystanie całego budżetu na innowacje zamiast stosowania podejścia zupełnie pozbawionego ryzyka.

Jak przełożyć to na praktyczny proces oceny ryzyka? Zacznijmy od tego, czego *nie robić*!

Czego nie robić?

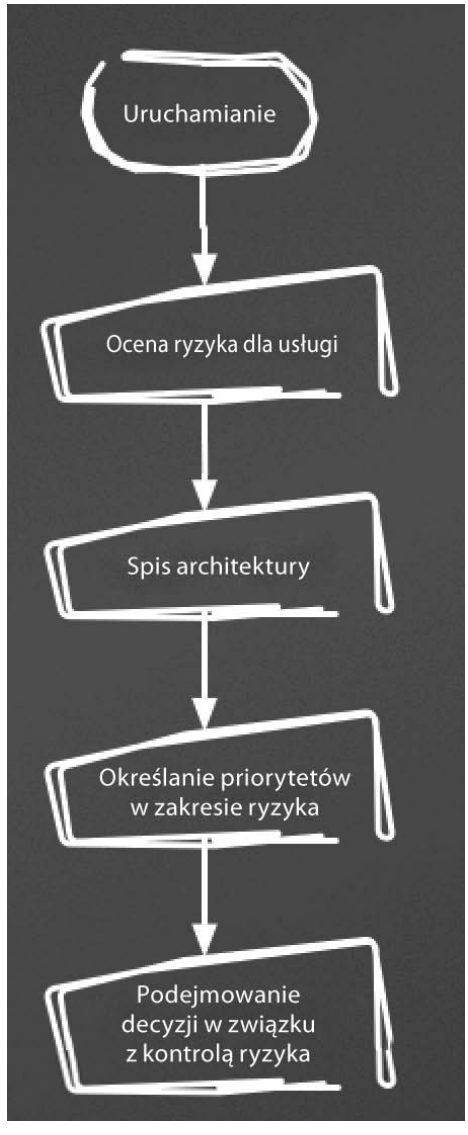
Trzeba pamiętać o wielu kwestiach. Oto kilka ostatnich wskazówek, które należy rozważyć w procesie zarządzania ryzykiem.

- Nie pozwól na to, by subiektywne przekonania zaszkodziły procesowi.
- Nie pozwól na to, by historyjki i plotki były głównym źródłem oceny ryzyka.
- Nie koncentruj się wyłącznie na dawnych incydentach i problemach. Patrz w przyszłość.
- Nie popadaj w stagnację. Analizuj wcześniejsze dokonania.
- Nie ignoruj czynnika ludzkiego.
- Nie ignoruj zmian w architekturze i przepływie pracy.
- Nie zakładaj, że obecne środowisko jest takie samo jak wcześniejsze.
- Nie twórz delikatnych mechanizmów kontrolnych i nie ignoruj najgorszych scenariuszy.

Jesteśmy przekonane, że z czasem dodasz do tej listy nowe pozycje. Jest to jednak dobry wykaz, o którym warto pamiętać, aby uniknąć pułapek, jakie czekają na Ciebie w trakcie analizowania systemów.

Funkcjonujący proces — uruchamianie

Niezależnie od tego, czy proces ma dotyczyć nowej, czy odziedziczonej usługi, zaczyna się od początkowego uruchomienia. W trakcie uruchamiania procesu (zob. rysunek 3.1) celem jest zidentyfikowanie głównych niebezpieczeństw, które mogą zagrozić spełnieniu celów SLO stawianych usłudze i których wystąpienie jest najbardziej prawdopodobne. Ponadto należy uwzględnić najgorsze scenariusze, które mogą zagrozić powodzeniu usługi w dłuższej perspektywie. Pamiętaj, dzielny inżynierze baz danych, że celem nie jest tu próba opracowania kompletnego spisu zagrożeń. Zadanie polega na utworzeniu listy wstępnej na potrzeby łagodzenia i eliminowania problemów oraz planowania tego, jak — operacyjnie — zapewnić największą wartość za pomocą dostępnych obecnie zasobów.



Rysunek 3.1. Początkowe uruchamianie procesu zarządzania ryzykiem

Ocena ryzyka dla usługi

Mając listę usług i mikrousług, którymi się opiekujesz, powinieneś usiąść z właścicielami produktu i ocenić tolerancję ryzyka dla każdej z nich. Oto pytania, na które warto odpowiedzieć.

- Jakie cele SLO z zakresu dostępności i opóźnienia są zdefiniowane dla danej usługi, gdy:
  - wszyscy użytkownicy są dotknięci;
  - dotknięty jest podzbiór użytkowników;

- usługa działa w trybie awaryjnym (tylko do odczytu, z wyłączonymi niektórymi funkcjami itd.);
- wydajność usługi jest obniżona.

- Jaki jest koszt przestoju danej usługi?
  - Utrata przychodów?
  - Utrzymanie klientów?
    - Usługa jest płatna czy darmowa?
    - Czy działa konkurencja, do której klient może łatwo przejść?
  - Czy przestoje mają skutki, które mogą wpłynąć na całą firmę?
    - Utrata danych?
    - Naruszenie prywatności?
    - Przestój w trakcie jakiegoś wydarzenia lub dni wolnych od pracy?
    - Przedłużający się przestój?

Przyjrzyjmy się przykładowi. UberPony to firma udostępniająca kucyki na żądanie. W jej serwisie działa sześć usług.

1. Rejestrowanie nowego klienta.
2. Składanie i realizacja zamówień kucyków na żądanie.
3. Rejestrowanie opiekunów kucyków.
4. Logistyka dla opiekunów kucyków.
5. Obsługa płatności dla opiekunów kucyków.
6. Wewnętrzne mechanizmy analityczne.

Przyjrzyjmy się teraz dwóm usługom — rejestrowaniu nowych klientów oraz składaniu i realizacji zamówień.

UberPony — rejestrowanie klientów	
Cel SLO — dostępność	99,90%
Cel SLO — opóźnienie	1 sekunda
Nowi klienci dziennie	5000
Cel SLO — dozwolone błędy	5
Koszt infrastruktury dziennie	13 698 zł
Koszt infrastruktury na złotówkę przychodów	0,003 zł
Wartość życiowa klienta	1 000 zł
Wartość życiowa dziennie	5 000 000 zł
Szczytowa liczba klientów na minutę	100
Odsetek klientów rezygnujących po błędzie	60%
Szczytowy poziom utraty wartości na minutę	60 000 zł

UberPony — składanie i realizacja zamówień	
Cel SLO — dostępność	99,90%
Cel SLO — opóźnienie	1 sekunda
Liczba zamówień dziennie obecnie	500 000
Cel SLO — dozwolone błędy	500
Koszt infrastruktury dziennie	30 000 zł
Koszt infrastruktury na złotówkę przychodów	0,006 zł
Przychód na zamówienie	10 zł
Przychód dzienny	5 000 000 zł
Szczytowa liczba zamówień na minutę	1000
Odsetek rezygnacji z zamówienia po błędzie	25%
Odsetek utraconych klientów po błędzie	1%
Szczytowa utrata przychodów na minutę	2500 zł
Utrata wartości generowanej przez klienta na minutę	10 000 zł
Łączna utrata na minutę	12 500 zł

Wygląda więc na to, że usługa rejestrowania klientów może kosztować aż do 4,8 razy więcej przychodów na minutę niż usługa składania i realizacji zamówień. 75% klientów ponowi zamówienie, jednak tylko 40% wróci, jeśli nie będzie mogło się zarejestrować. Najwyraźniej chętnie skorzystają z usługi UberDonkey. Zauważ, że próbowaliśmy uwzględnić zmienne, takie jak utrata klientów po błędzie w trakcie składania zamówienia, a także odsetek ponawianych prób rejestracji i złożenia zamówienia po wystąpieniu problemu. Ustalenie tych wartości bez solidnych danych analitycznych może być trudne, jeśli jednak takie informacje są niedostępne, wystarczające mogą być szacunki. Lepsze to niż nic.

Dane się zmieniają, dlatego koniecznie aktualizuj dane, gdy stosujesz opisany proces. Jeśli np. firma UberDonkey stanie się bardziej konkurencyjna i UberPony zacznie tracić 5% klientów po błędzie w czasie składania zamówienia, koszty minuty przestoju w usłudze składania i realizacji zamówień nagle wzrosną do 52 500 zł. To zdecydowanie zwiększy priorytet tej usługi. Jednak obecnie znacznie sensowniejsze jest skupienie się na usłudze rejestrowania klientów.

Spis architektury

Po zdefiniowaniu zakresu prac należy spisać systemy i środowiska, za które odpowiadasz. Uwzględnij:

- centra danych;
- komponenty i warstwy architektury (np. MySQL, równoważniki obciążenia Nginx, instancje aplikacji J2EE, sieć, zapory, Hadoop/HDFS, sieć CDN);
- role przydzielone tym komponentom (np. węzeł zapisujący/główny, replika);
- ścieżki interakcji i komunikacji między usługami (zapytania z aplikacji do bazy MySQL czy z równoważnika obciążenia do aplikacji, przesyłanie danych przez aplikację do bazy Redis);
- zadania (proces pobierania, przekształcania i wczytywania, ang. *extract, transform, load* — ETL, wczytywanie danych z sieci CDN, odświeżanie pamięci podręcznej, zarządzanie konfiguracją, koordynacja, tworzenie kopii zapasowych i odzyskiwanie danych, agregacja dzienników).

Oto uproszczony spis dotyczący priorytetowej usługi.

UberPony — rejestrowanie klientów		
Komponent	***Liczba w centrum danych 1***	***Liczba w centrum danych 2***
Równoważniki obciążenia frontonu	2	2
Serwery WWW	20	20
Równoważniki obciążenia dla Javy	2	2
Serwery Javy	10	10
Serwery pośredniczące bazy danych	2	2
Sieć CDN CloudFront	Usługa	Usługa
Serwery z pamięcią podręczną bazy Redis	4	4
Serwery zapisujące klastra z bazą MySQL	1	1
Serwery odczytujące klastra z bazą MySQL	2	2
Replikacja bazy MySQL	Usługa	Usługa
Odświeżanie zawartości sieci CDN	Zadanie	Zadanie
Odświeżanie pamięci podręcznej bazy Redis	Zadanie	Zadanie
Tworzenie kopii zapasowych bazy MySQL	Zadanie	Brak
Proces ETL	Zadanie	Brak
Hurtownia danych RedShift	Usługa	Brak

Następny krok polega na ocenie w tej architekturze zagrożeń, które mogą wpływać na usługę.

Priorytety

Jak zidentyfikować zagrożenia, które mogą spowodować naruszenie celów SLO, i ustalić priorytety tych niebezpieczeństw? W ramach zarządzania ryzykiem zagrożenia są definiowane w kategoriach prawdopodobieństwa spowodowania niekorzystnych skutków przez dany problem pomnożonego przez konsekwencje danej sytuacji. W tabeli poniżej pokazano spektrum możliwości.

Prawdopodobieństwo/ skutki	**Bardzo poważne**	**Poważne**	**Umiarkowane**	**Niewielkie**	**Pomijalne**
Prawie pewne	Nieakceptowalne	Nieakceptowalne	Wysokie	Umiarkowane	*Akceptowalne*
Prawdopodobne	Nieakceptowalne	Wysokie	Wysokie	Umiarkowane	*Akceptowalne*
Możliwe	Nieakceptowalne	Wysokie	Umiarkowane	Umiarkowane	*Akceptowalne*
Mało prawdopodobne	Wysokie	Umiarkowane	Umiarkowane	Akceptowalne	*Akceptowalne*
Rzadkie	Wysokie	Umiarkowane	Akceptowalne	Akceptowalne	*Akceptowalne*

W celu wyeliminowania niejasności ważne jest, aby liczbowo określić poziomy prawdopodobieństwa i skutków. Skutki zmieniają się w zależności od dziedziny problemu. Niejasności w kwestii prawdopodobieństwa zostały dobrze opisane w artykule *Describing probability: The limitations of natural language* (*http://www.risk-doctor.com/pdf-files/emeamay05.pdf*).

Ustalmy poziomy prawdopodobieństwa w następujący sposób.

Skala	Przedział
Prawie pewne	>50%
Prawdopodobne	26 – 50%
Możliwe	11 – 25%
Mało prawdopodobne	5 – 10%
Rzadkie	<5%

Uznajmy, że jest to procent czasu naruszania celów SLO w danym okresie, np. tygodniowym. Jeśli chodzi o skutki, to uwzględniamy cele SLO w ramach określania ich kategorii, a także w trakcie analizy innych problemów, które grożą upadkiem firmy (są to np. uszkodzenie danych, naruszenie prywatności i incydenty związane z bezpieczeństwem). Większość takich problemów należy do kategorii bardzo poważne lub poważne. Warto przypomnieć, że tu przedstawiamy tylko przykłady.

Bardzo poważne skutki (natychmiastowe naruszenie celu SLO)

Oto definicja bardzo poważnych skutków.

- Cała usługa staje się niedostępna lub działa z opóźnieniem powyżej 100 milisekund przez przynajmniej 10 minut dla 5% lub więcej użytkowników. W tygodniu jest 10 080 minut, dlatego 10 minut przestoju narusza cel SLO mówiący o 99,9% dostępności.
- Zbliżające się lub mające już miejsce ujawnienie danych jednych klientów innym.
- Pozwolenie nieupoważnionym osobom na dostęp do systemów i (lub) danych produkcyjnych.
- Uszkodzenie danych transakcyjnych.

Każdy z tych punktów oznacza przypisanie problemów do kategorii bardzo poważne.

Poważne (zbliżające się naruszenie celów SLO)

Oto cechy poważnych skutków.

- Cała usługa jest niedostępna lub działa z opóźnieniem powyżej 100 milisekund przez trzy do pięciu minut dla przynajmniej 5% użytkowników (aż do zużycia 50% budżetu niedostępności).
- Spadek wydajności systemu do 100% wymaganej (z docelowych 200%).

Każdy z tych punktów oznacza przypisanie problemów do kategorii poważne.

Umiarkowane (może skutkować naruszeniem celów SLO, jeśli w tym samym okresie wystąpią też inne incydenty)

Oto cechy umiarkowanych skutków.

- Cała usługa jest niedostępna lub działa z opóźnieniem powyżej 100 milisekund przez minutę do trzech minut dla przynajmniej 5% użytkowników (aż do zużycia 33% budżetu niedostępności).
- Spadek wydajności systemu do 125% wymaganej (z docelowych 200%).

Każdy z tych punktów oznacza przypisanie problemów do kategorii umiarkowane.

Niewielkie

Oto cechy niewielkich skutków.

- Cała usługa jest niedostępna lub działa z opóźnieniem powyżej 100 milisekund przez okres do minuty dla przynajmniej 5% użytkowników (aż do zużycia 10% budżetu niedostępności).
- Spadek wydajności systemu do 150% wymaganej (z docelowych 200%).

Każdy z tych punktów oznacza przypisanie problemów do kategorii niewielkie.

Warto przypomnieć, że nie próbujemy uwzględnić każdego możliwego zagrożenia. Ten spis będzie stale rozwijany w wyniku zarządzania incydentami i procesów zarządzania ryzykiem. Tworzymy tu tylko *schemat* o ograniczonym zakresie, aby zapewnić podstawy do pragmatycznych działań. W tym scenariuszu budujemy schemat na podstawie scenariuszy, które są *najbardziej prawdopodobne* i *mają najpoważniejsze skutki*.

Wiemy np., że awarie komponentów i instancji są częste w publicznych chmurach, takich jak ta, której firma UberPony używa do hostingu. Oznacza to, że występuje w nich niski średni czas między awariami (MTBF). Dla grup instancji serwerów WWW i Javy kategoryzujemy takie awarie jako „prawdopodobne", ponieważ cały czas korzystamy z umiarkowanej liczby takich instancji (20 lub 10). Oznacza to, że awaria jednej instancji serwera WWW dotyka 5% klientów. Z kolei awaria instancji Javy wpływa na 10% klientów. Wynikiem jest naruszenie celów SLO, a ponieważ uruchomienie nowej instancji może zająć od trzech do pięciu minut, skutki są *poważne*. Dla prawdopodobieństwa na poziomie *prawdopodobne* i skutków na poziomie *poważne* ryzyko jest *wysokie*. Po wprowadzeniu zautomatyzowanych działań naprawczych (wycofywania danej instancji z usługi i uruchamiania nowej) przeprowadzamy testy i okazuje się, że ten proces zajmuje średnio pięć sekund. Poziom skutków zmienia się wtedy na *niewielkie*, a tym samym kategoria ryzyka ulega zmianie na *umiarkowane*.

Jeśli przeanalizujesz awarie na poziomie usługi lub instancji na podstawie spisu, możesz uzyskać następującą listę.

UberPony — usługa rejestrowania klientów

Komponent	*Prawdopodobieństwo*	*Skutki*	*Ryzyko*
Równoważniki obciążenia frontonu	Możliwe	Bardzo poważne	Nieakceptowalne
Serwery WWW	Prawdopodobne	Poważne	Wysokie
Równoważniki obciążenia dla Javy	Możliwe	Poważne	Wysokie
Serwery Javy	Prawdopodobne	Poważne	Wysokie
Serwery pośredniczące bazy danych	Możliwe	Poważne	Wysokie
Sieć CDN CloudFront	Rzadkie	Poważne	Umiarkowane
Serwery z pamięcią podręczną bazy Redis	Możliwe	Poważne	Umiarkowane
Serwery zapisujące klastra z bazą MySQL	Mało prawdopodobne	Bardzo poważne	Wysokie
Serwery odczytujące klastra z bazą MySQL	Możliwe	Poważne	Wysokie
Replikacja bazy MySQL	Możliwe	Poważne	Wysokie
Odświeżanie zawartości sieci CDN	Mało prawdopodobne	Niewielkie	Akceptowalne
Odświeżanie pamięci podręcznej bazy Redis	Mało prawdopodobne	Niewielkie	Akceptowalne
Tworzenie kopii zapasowych bazy MySQL	Mało prawdopodobne	Niewielkie	Akceptowalne
Proces ETL	Mało prawdopodobne	Niewielkie	Akceptowalne
Hurtownia danych RedShift	Rzadkie	Niewielkie	Akceptowalne

Na podstawie takiego schematu należy w pierwszej kolejności dokładniej zająć się wszystkimi komponentami z ryzykiem z poziomu *nieakceptowalne* i *wysokie*, potem przejść do komponentów o *umiarkowanym* ryzyku itd. Po omówieniu podstaw eksploatacji, w punkcie poświęconym bazom danych, przeprowadzimy szczegółową analizę ryzyka dla baz danych. Tu celem jest ułatwienie zrozumienia procesu. Inna uwaga dotyczy tego, że trzeba też uwzględnić zagrożenia z poziomu całego centrum danych. Choć takie problemy są rzadkie, należą do tej samej kategorii, co naruszenie prywatności, utrata danych i inne niebezpieczeństwa wymagające przemyślenia z powodu skutków mogących zagrozić przetrwaniu firmy.

Mechanizmy kontrolne i podejmowanie decyzji

Po określeniu priorytetów na podstawie listy analizowanych zagrożeń przyjrzymy się technikom decydowania o mechanizmach kontrolnych pozwalających łagodzić, a czasem nawet eliminować niebezpieczeństwa. Zaczęłyśmy omawianie tego tematu już w poprzednim punkcie, wprowadzając automatyczne zastępowanie serwerów WWW i Javy, aby skrócić średni czas do przywrócenia stanu (MTTR) po awarii. Pamiętaj, że koncentrujemy się na błyskawicznym przywracaniu stanu i skracaniu MTTR, a nie na unikaniu awarii. Ważniejsza jest odporność niż krucha wysoka dostępność!

Dlaczego przedkładamy MTTR nad MTBF?

Gdy tworzysz system, który rzadko się psuje, będzie on z natury kruchy. Czy zespół będzie gotowy do naprawy systemu po awarii? Czy w ogóle będzie wiedział, co ma robić? W systemach, w których awarie są częste, ale kontrolowane i naprawiane w taki sposób, by ich skutki były pomijane, wiadomo, co zrobić, gdy coś się zepsuje. Procesy są dobrze udokumentowane i pielęgnowane, a zautomatyzowane działania naprawcze stają się przydatne, zamiast ukrywać się w mrocznych zakamarkach systemu.

Dla każdego możliwego zagrożenia zespół może wybrać jedno z trzech podejść:

- unikanie (znalezienie sposobu na wyeliminowanie zagrożenia);
- ograniczanie skutków (ustalenie, jak zmniejszyć wpływ problemu, gdy już wystąpi);
- akceptacja (uznanie zagrożenia za akceptowalne i odpowiednie zaplanowanie reakcji na nie).

W technicznym sensie w kręgach specjalistów od zarządzania ryzykiem istnieje też czwarte podejście — podział ryzyka za pomocą outsourcingu, wykupienia ubezpieczenia lub innych metod przeniesienia ciężaru na kogoś innego. Żadna z tych metod nie ma jednak zastosowania do ryzyka w branży informatycznej, dlatego nie będziemy tu analizować tego podejścia.

Każdy komponent należy przeanalizować pod kątem rodzajów awarii, ich skutków i zestawu mechanizmów kontrolnych pozwalających na automatyczne przywracanie stanu, skrócenie czasu przywracania stanu i zmniejszenie częstotliwości występowania problemu. Określane są też koszty i wysiłek związane z tymi mechanizmami. W wyniku ich porównania z kosztami przestoju można podjąć decyzje dotyczące właściwego sposobu radzenia sobie z zagrożeniami.

Identyfikowanie

W trakcie analizy ryzyka w firmie UberPony zidentyfikowaliśmy kilka warstw usług bazodanowych MySQL cechujących się wysokim ryzykiem. Jest to bardzo typowe w warstwie bazy danych.

Zobaczmy teraz, co można zrobić, aby ograniczyć to ryzyko. W usłudze wykryliśmy cztery główne rodzaje awarii:

- awaria instancji zapisującej,
- awaria instancji wczytującej,
- awaria replikacji,
- awaria tworzenia kopii zapasowych.

Są to rodzaje awarii często występujące w magazynach danych.

Ocena

Jeśli chodzi o awarie zapisu, zespół ds. eksploatacji w UberPony ocenia możliwości automatyzacji przywracania stanu po awarii serwera zapisującego bazy MySQL. Po wystąpieniu takiej awarii usługa rejestrowania klientów nie może tworzyć ani modyfikować żadnych danych. To oznacza brak nowych klientów i brak możliwości modyfikowania ich danych przez nich samych lub przez firmę. Stwierdziliśmy, że jeśli usługa rejestrowania przestanie działać przy szczytowym obciążeniu, możemy tracić 60 000 zł wartości życiowej klienta na minutę. Dlatego bardzo ważne jest rozwiązanie tego problemu. Oznacza to, że *akceptacja ryzyka* nie wchodzi w grę.

Łagodzenie skutków i mechanizmy kontrolne

Gotowe są już pewne mechanizmy *eliminowania ryzyka*. Używany jest 10-dyskowy system RAID, który zapewnia nadmiarowe dyski. Dlatego awaria dysku nie prowadzi do awarii bazy. W środowisku występują też inne podobne nadmiarowe komponenty. Inna proponowana technika *eliminowania* to zastąpienie podstawowego silnika MySQL-a narzędziem Galera, którego architektura umożliwia zapis danych w dowolnym węźle klastra bazy MySQL. To wymaga znacznych zmian architektury, a nikt w zespole nie ma dużego doświadczenia w korzystaniu z Galery. Po zastanowieniu okazuje się, że zagrożenia powodowane przez nowy system zdają się przeważać nad korzyściami z jego wprowadzenia.

Gdyby odpowiednio zaprojektować aplikację, klienci nadal mogliby logować się do usługi i wyświetlać dane z instancji odczytujących. Na tym polega *łagodzenie skutków zagrożeń*. Po rozmowie z inżynierami oprogramowania okazuje się, że planują wprowadzić takie rozwiązanie. Jednak w trybie awaryjnym nowi klienci i tak nie mogą się rejestrować, dlatego ten mechanizm nie zapewnia firmie istotnych korzyści (rezygnacja z rozwijania nowych funkcji to poważny koszt na konkurencyjnym rynku).

Ostatecznie zespół decyduje się zastosować automatyczne działania naprawcze. Tu polegają one na automatycznym przełączaniu awaryjnym na inny serwer główny. Zespół wybiera przełączanie awaryjne zamiast ręcznego, ponieważ 10 minut przestojów dopuszczalne w celach SLO nie wystarcza, aby człowiek zdążył wykonać odpowiednie czynności. W trakcie zarządzania zapisem może nastąpić utrata danych, dlatego proces musi działać bezbłędnie.

Implementacja

Zespół decyduje się zastosować technologię MySQL MHA do automatycznego przełączania awaryjnego. MySQL MHA (ang. *MySQL High Availability*) to oprogramowanie do zarządzania przełą-

czaniem awaryjnym i wprowadzania zmian w topologii replikacji potrzebnych w wyniku takiego przełączania. Zespół opracowuje plan dokładnych testów przed zaimplementowaniem tak ważnego procesu. Testy są wykonywane etapowo — początkowo w środowisku testowym bez ruchu, potem w środowisku testowym z symulowanym ruchem, a ostatecznie w środowisku produkcyjnym, gdzie są ściśle monitorowane. Testy są przeprowadzane wielokrotnie, aby mieć pewność, że dany wynik nie jest wyjątkiem. Oto, co obejmują testy.

- Kontrolowane zamykanie głównej bazy danych w środowisku testowym.
- Zamykanie procesu MySQL w środowisku testowym.
- Zamykanie w środowisku testowym instancji serwera, na którym działa MySQL.
- Symulowanie podziału sieci.

Po każdym teście zespół wykonuje następujące czynności.

- Rejestrowanie czasu, jaki zajęło przełączanie awaryjne.
- Rejestrowanie opóźnienia dla symulowanego i produkcyjnego ruchu, aby ocenić wpływ procesu na wydajność.
- Upewnianie się, że tabele nie zostały uszkodzone.
- Upewnianie się, że dane nie zostały utracone.
- Sprawdzanie dzienników błędów u klientów, aby zobaczyć wpływ procesu.

Gdy zespół jest usatysfakcjonowany tym, że system działa i spełnia cele SLO, zastanawia się, jak włączyć nowy proces w inne standardowe czynności. Wymaga to upewnienia się, że proces jest dobrze przećwiczony, udokumentowany i wolny od błędów. Zespół początkowo decyduje się powiązać nowe działania z procesem wdrażania i wykorzystać przełączanie awaryjne do wprowadzania stopniowych zmian w obiektach bazodanowych, aby nie zakłócać jednowątkowego procesu replikacji baz MySQL. Gdy zespół kończy pracę, przełączanie awaryjne zajmuje nie więcej niż 30 sekund.

W trakcie sprawdzania procesu przełączania awaryjnego inżynierowie oprogramowania odkryli też, że w czasie 30 sekund, jakie zajmuje to zadanie, może nastąpić utrata danych. Dlatego zespół wprowadza podwójny zapis w aplikacjach. Oznacza to, że wszystkie operacje wstawiania, aktualizacji i usuwania danych są kierowane do brokera zdarzeń na wypadek, gdyby trzeba było je odtworzyć. Są to dodatkowe działania łagodzące skutki przełączania awaryjnego głównego serwera zapisującego.

Te mechanizmy kontrolne tworzy się na potrzeby wstępnego uruchamiania programu. Ważne jest, by pamiętać, że nie muszą być idealne. Jest to tylko początkowy zestaw mechanizmów, powiązanych z najbardziej priorytetowymi kwestiami i generujących największą wartość.

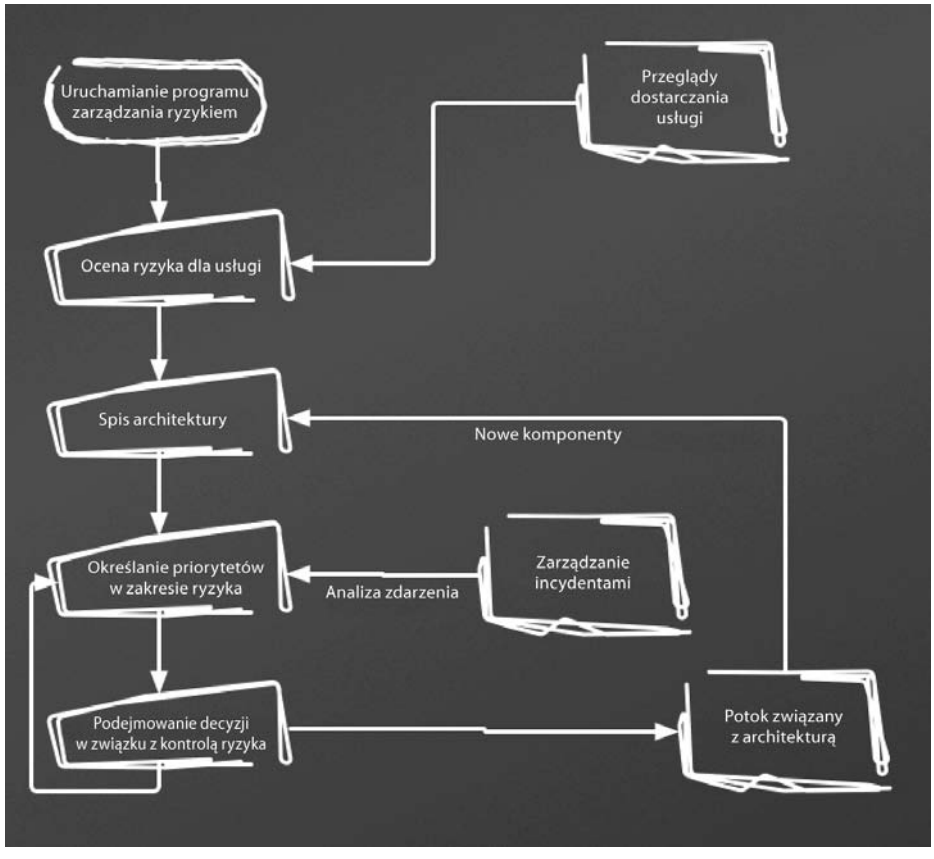
Opracowanie początkowej wersji procesu oznacza, że przeszedłeś już długą drogę w kierunku uwzględnienia najczęstszych źródeł ryzyka. Od tego momentu prace przebiegają iteracyjnie.

Bieżące iteracje

Gdy początkowy proces jest już gotowy, priorytetem w zakresie eliminowania i łagodzenia skutków zagrożeń staje się związany z architekturą potok projektowania, budowania i bieżącej eksploatacji rozwiązań. Wspomniałyśmy wcześniej, że zarządzanie ryzykiem to proces ciągły. Nie trzeba

więc od razu uwzględniać wszystkiego, ponieważ realizowane procesy pozwalają rozbudować listę zagrożeń i zapewnić lepsze pokrycie systemu, co ilustruje rysunek 3.2. Oto te procesy.

- Przeglądy dostarczania usługi.
- Zarządzanie incydentami.
- Potok architektoniczny.



Rysunek 3.2. Stale powtarzający się cykl i dane wejściowe w procesie zarządzania ryzykiem

Przeglądy dostarczania usługi to okresowe analizy ewolucji usługi z naciskiem na zmiany w tolerancji ryzyka, przychodach, kosztach skutków i bazy użytkowników. Gdy te aspekty znacznie się zmieniają, trzeba ponownie przyjrzeć się wcześniejszym poziomom akceptacji ryzyka, działaniom naprawczym i technikom eliminowania zagrożeń, aby mieć pewność, że nadal są one akceptowalne.

Procesy zarządzania incydentami zapewniają dane wejściowe do określania priorytetów. Gdy analiza zdarzenia ujawnia nowe luki, trzeba się im przyjrzeć i uwzględnić je na liście priorytetów. Ponadto w ramach budowania potoku architektonicznego trzeba wprowadzić do procesu zarządzania ryzykiem nowe komponenty, aby zidentyfikować zagrożenia, które mogły zostać pominięte na etapie projektowania.

Podsumowanie

Poznałeś już znaczenie włączania zarządzania ryzykiem w codzienne procesy informatyczne. Zapoznałeś się też z różnymi zagadnieniami i czynnikami, które mogą wpływać na proces zarządzania ryzykiem, oraz przeanalizowałeś realistyczny model uruchamiania zarządzania ryzykiem w połączeniu z codziennymi procesami, co pozwala z czasem stopniowo rozwijać ten proces.

Jednak nawet po zrozumieniu zobowiązań związanych z poziomem jakości usług i potencjalnych zagrożeń w ich realizacji brakuje bardzo istotnego elementu, czyli monitorowania operacyjnego. Wgląd w obecną sytuację, a także wiedza na temat wcześniejszej wydajności i charakterystyki systemu będą potrzebne, aby zapobiegać problemom i podejmować decyzje związane z tym, jak stale usprawniać system, którym zarządzasz.

Skorowidz

E

F

H

I

J

K

L

M

N

O

P

R

S

Ś

T

U

W

Z

O'REILLY®

Poznaj zasady inżynierii niezawodności!

Rewolucja informatyczna dosięgła również systemów bazodanowych. Przez długi czas administrator bazy danych interesował się głównie wewnętrznymi mechanizmami bazy, optymalizacją zapytań czy analizą podsystemów składowania danych. Z kolei oprogramowaniem stron, infrastrukturą i usługami sieciowymi zajmowali się zupełnie inni ludzie, pracujący w odmienny sposób. Nowe technologie wymuszają jednak zmianę sposobu pracy i myślenia. Trzeba położyć nacisk na automatyzację, inżynierię oprogramowania, ciągłą integrację i ciągłe udostępnianie. Poza tym trzeba zapewnić ochronę przetwarzanych danych – ich wartość i znaczenie wciąż szybko rosną.

W tej praktycznej książce dokładnie wyjaśniono współczesne podejście do tworzenia architektury baz danych i ich eksploatacji. Jeśli chcesz być znakomitym inżynierem niezawodności baz danych, czyli DBRE (*Database Reliability Engineer*), znajdziesz tu schemat zasad oraz praktyk projektowania, budowania i eksploatacji magazynów danych zgodnie z paradygmatami inżynierii niezawodności i kultury DevOps. Zapoznasz się z podstawowymi zagadnieniami z obszaru eksploatacji i metodami utrwalania baz danych, a także nauczysz się stosować najważniejsze technologie skalowalnego i wydajnego składowania oraz pobierania danych z zachowaniem odporności na błędy. Dzięki temu szybko i skutecznie zajmiesz się architekturą i eksploatacją każdej nowoczesnej bazy.

W tej książce między innymi:

- wprowadzenie do inżynierii niezawodności baz danych
- inżynieria infrastruktury i zarządzanie nią
- oceny ryzyka i strategie zarządzania bezpieczeństwem danych
- metody przechowywania, indeksowania i replikacji danych
- popularne wzorce architektoniczne rozproszonych baz danych

Laine Campbell – od 18 lat zajmuje się środowiskami produkcyjnymi baz danych i systemów rozproszonych o dużej skali. Obecnie jest starszym dyrektorem ds. inżynierii środowisk produkcyjnych w firmie Fastly.
Charity Majors – jest CEO i założycielką firmy Honeycomb.io. Wcześniej zajmowała się eksploatacją należącej do Facebooka platformy Parse – zarządzała rozbudowanym zestawem replik baz MongoDB, a także bazami Redis, Cassandra i MySQL.

ISBN 978-83-283-4426-6
9 788328 344266

Cena: 59,00 zł