Zastosowanie baz danych noSQL w grach komputerowych

Michał Rokicki

TouK
al. Bohaterów Września 9 02-389 Warszawa, www.touk.pl
michal.rokicki@gmail.com

Streszczenie

Artykuł dotyczy zastosowania baz danych noSQL, jako alternatywy wobec baz relacyjnych w grach online. Omówiono właściwości baz najczęściej używanych przez twórców gier. Na przykładzie Redis, MongoDB oraz CouchDB, przedstawiono ich wady i zalety, oraz podano rozwiązania stosowane w istniejących produkcjach. Ponadto przeprowadzono analizę wymagań gier komputerowych wobec oprogramowania, na tej podstawie przedstawiono powody, dla których warto stosować nowe technologie bazodanowe.

1. Wstęp

Współcześnie produkuje się coraz więcej gier online. Gry tego typu przechowują dane po stronie serwera, muszą więc wykorzystywać systemy

bazodanowe które spełniają ich potrzeby.
Po pierwsze gry online zazwyczaj nie przynoszą dużego dochodu w przeliczeniu na gracza, generują za to duży ruch. Sprawia to, że producenci muszą szukać tanich i szybkich rozwiązań. Drugim problemem jest skalowalność, istnieje dużo gier, dla których liczba graczy liczona jest w milionach i ciągle rośnie. Obciążenie, jakiemu serwer bazy danych jest poddany, przerasta wówczas możliwości nawet najpotężniejszej maszyny. Jedynym rozwiązaniem jest rozproszenie systemu na wiele maszyn.

Oba wymienione problemy nie znajdują rozwiązania na gruncie baz relacyjnych, które są powolne i nie skalują się automatycznie. Oprogramowanie skalowania prowadzi do znacznego spadku wydajności, lub do utraty części funkcjonalności (np. ACID). W wielu wypadkach najlepszym rozwiązaniem jest zastosowanie bazy noSQL.

2. Czym jest noSQL

Bazy noSQL to systemy nieoparte na tradycyjnym modelu relacyjnym. Nie zawsze implementują wszystkie cechy ACID (Atomicity – atomowość, Consistency – spójność, Isolation – izolacja, Durability – trwałość), dane zazwyczaj nie są zorganizowane w tabelach, nie pozwalają na stosowanie złączeń, często łatwo skalują się wszerz, pozwalają na gromadzenie ogromnych ilości danych oraz odznaczają się wysoką wydajnością. Nie jest to jedna technologia, a raczej cała klasa systemów bazodanowych stojących w opozycji do modelu relacyjnego. Obecnie uznaje się, że skrót noSQL rozwija się jako not only SQL.[1]

2.1 Klasyfikacja baz noSQL

- **Klucz–wartość –** bazy te pozwalają na dostęp do wartości będącymi pojedynczymi obiektami poprzez klucze. Przykłady: Redis, Membase, Amazon Dynamo.

- **Kolumnowe** – pozwalają na mapowanie za pomocą klucza zbioru kolumn o określonych typach. Przykłady: Google BigTable, Cassandra.

- **Dokumentowe** – te bazy indeksują za pomocą kluczy dokumenty. Dokumenty to zbiory wartości klucz-wartość o strukturze drzewiastej. Podejście takie jest niezwykle elastyczne i pozwala na łatwe odwzorowanie danych. Przykłady: CouchDB, MonogDB.

- Pozostałe: **Grafowe** – dane zorganizowane są w strukturze grafowej. **XML** – bazy traktujące plik w formacie xml jako źródło danych. W momencie gdy niektóre bazy relacyjne wprowadziły typ kolumny XML i umożliwiło obsługę xPathów, rozwiązanie to stało się mało popularne.[2] **Obiektowe** – bazujące na modelu obiektowym. W praktyce jednak okazały się mało wydajne. (Te typy baz mają małe zastosowanie przy tworzeniu gier).

3. Pojęcia kluczowe

- **Sharding** – z angielskiego skrót od „shared-nothing". Jest to sposób rozdziału danych w systemie bazodanowym pomiędzy wiele maszyn w taki sposób, by żadne dane nie były współdzielone [3]. Zazwyczaj pozwala to zwiększyć szybkość działania bazy danych.

- **Failover** – automatyczne przełączanie kopii systemu, w przypadku gdy poprzednio działająca instancja ulegnie awarii. To podejście sprawia, że awaria pojedynczej maszyny nie powoduje przerwania działania całego systemu [4].

- **Skalowanie wszerz** – rozbudowa systemu poprzez dodanie kolejnych maszyn zajmujących się jego obsługą [2].

- **skalowanie wzwyż** – polega na podniesieniu wydajności serwera (na przykład poprzez zainstalowanie lepszego procesora, wymiany dysków itp.) [2].

4. Wady i zalety modelu relacyjnego

Model relacyjny stosowany jest z powodzeniem od lat siedemdziesiątych, głównie przy oprogramowaniu biznesowym. W grach komputerowych relacyjne bazy danych zaczęły być stosowane wraz z pojawieniem się gier online.

Zalety

- Idealnie nadają się do usystematyzowanych danych (np. księgowych)

- Gwarantują ACID.

- Nie tylko gromadzą dane, również w znacznym stopniu pilnują ich struktury, poprzez udostępnienie kluczy obcych, różnego typu więzów, wyzwalaczy

- Normalizacja pozwala na uniknięcie redundancji danych

- W wielu wypadkach ich wydajność jest zadowalająca

- Istnieje wiele narzędzi do administracji relacyjnymi bazami danych

- Bazy relacyjne funkcjonują od 40 lat, wiele systemów bazodanowych jest rozwijanych od właśnie takiego czasu (np. Oracle[5]). Można więc uznać, że jest to bardzo dojrzała i niezawodna technologia.

Wady

- Mała elastyczność tabel jako źródeł danych, często tabela posiada bardzo wiele kolumn z pośród których tylko część jest istotna w danym rekordzie.

- Normalizacja zazwyczaj powoduje duże problemy z wydajnością.

- Bazy relacyjne są praktycznie nieskalowalne wszerz. Rozproszenie bazy relacyjnej zazwyczaj powoduje duży spadek wydajności lub wymaga rezygnacji z pełnego ACIDa.

- Wprowadzanie zmian w strukturze tabel może być problematyczne. Dotyczy to szczególnie bazy danych z dużą liczbą wierszy, w których np. dodanie nowej kolumny może trwać bardzo długo.

4.1 Porównanie potrzeb oprogramowania biznesowego i gier komputerowych

Oprogramowanie biznesowe różni się w większości przypadków od gier komputerowych online pod względem wymagań. Zazwyczaj jego funkcjonowanie przynosi dochód lub wprowadza znaczne oszczędności dla przedsiębiorstwa które je stosuje. Koszty zakupu odpowiednio wydajnych serwerów oraz ich utrzymanie są rekompensowane przez korzyści. Z drugiej

strony od tego typu systemów wymaga się dużej niezawodności. Konsekwencje nieprawidłowego działania oprogramowania biznesowego mogą powodować znaczne koszty zarówno finansowe, jak i wizerunkowe. Jaskrawym przykładem mogą być tu systemy bankowe, które w żadnym wypadku nie mogą sobie pozwolić na błędną obsługę przelewów.

Dla tego typu systemów bazy relacyjne ze znormalizowanym schematem danych wydają się bardzo dobrym rozwiązaniem właśnie, ze względu na swoją niezawodność.

Gry komputerowe online w przeliczeniu na gracza przynoszą często niewielki dochód. Coraz popularniejszy jest model Free to Play [6] w którym podstawowa funkcjonalność gry jest za darmo, płatne są konta premium, dodatki, dodatkowe przedmioty itp. Nawet kultowy World of Warcraft zrezygnował z abonamentu dla początkujących graczy [7]. W wielu grach większość użytkowników nie korzysta z płatnych dodatków, w takim przypadku jedyny dochód jaki generują pochodzi z odsłon reklam.

Oczywiście to, że użytkownik nie musi płacić za grę, nie znaczy, że jej utrzymanie nie kosztuje [8]. Z tego względu twórcy gier muszą znacznie bardziej pilnować kosztów, niż twórcy biznesowi.

Jednocześnie od gier online nie wymaga się tak dużej niezawodności jak od oprogramowania biznesowego. Dobrym przykładem jest gra Minecraft (posiadająca wersje online), którą można było kupić już na etapie wersji alfa. Od początku istniało w niej bardzo dużo błędów. Mimo to sprzedaż gry rosła lawinowo i biła kolejne rekordy [9].

Biorąc to wszystko pod uwagę widać, że nie zawsze stosowanie baz relacyjnych stworzonych z myślą o oprogramowaniu biznesowym będzie się sprawdzać w grach online. W dalszej części artykułu przedstawione zostaną rozwiązania bazodanowe nastawione na specyficzne potrzeby gamedevu.

5. Relacyjne bazy danych stosowane jak noSQL

Często zdarza się, że twórcy gier stosują bazy relacyjne, ale nie wykorzystują w pełni ich funkcjonalności, ze względu na wydajność, lub z uwagi na to, że są rozproszone. Może to polegać, na częściowej denormalizacji schematu danych, rezygnacji z używania joinów, lub serializowaniu obiektów i zapisywaniu ich np. w blobie, zamiast odzwierciedlenia ich struktury w tabelach. W przypadku gdy system nie znajduje się na jednej maszynie stosowanie złączeń i transakcji może być w ogóle niemożliwe.

5.1 Rozwiązania Playfish

Jaskrawym przypadkiem tego zjawiska jest rozwiązanie firmy Playfish[8]. W bazie MySQL stosowany jest model danych w których obiekty są serializowane do blobów, dostęp do nich zapewniają odpowiednie

indeksy. Do tego oprogramowany został automatyczny sharding oraz failover.

W rezultacie stworzono system przypominający funkcjonalnością bazę noSQL.

5.2 Częściowa denormalizacja

Grą, która używa relacyjnej bazy danych i jednocześnie stosuje częściową denormalizacje jest Gizarma[10]. Część danych nie jest odzwierciedlona w tabelach, zamiast tego jest zserializowana i trzymana w pojedyńczych polach tabel. Przykładowo tabela jednostka, posiada pole rozkazy, w których znajduje się zserializowana kolejka rozkazów.

Zaletą takiego podejścia, jest zachowanie możliwości korzystania z narzędzi przeznaczonych dla relacyjnych baz danych oraz zwiększenie wydajności zapytań.

6. Baza noSQL typu klucz-wartość.

Ta klasa baz zostanie omówiona na przykładzie bazy Redis.

6.1 Baza Redis [11]

- Struktura bazy przypomina tabelę w której są tylko 2 pola, klucz i wartość
- Wartość może przyjmować następujące typy: string, zbiór, posortowany zbiór, lista, tablica asocjacyjna
- Ponieważ dane są przechowywane w pamięci, odczyt i zapis trwa nieporównywalnie krócej niż w przypadku baz przechowujących dane na dysku
- Nie nadaje się do przechowywania bardzo dużych ilości danych, ze względu na mały rozmiar pamięci RAM serwerów
- Trwałość danych zapewnia okresowy i przyrostowy zapis na dysk.
- Przy odpowiedniej konfiguracji obsługuje transakcje co gwarantuje niezawodność

- Obsługuje również automatyczną replikacje (master - slave).
- Redis sprawdza się najlepiej, gdy trzyma się w nim dużo informacji o niewielkich rozmiarach, które często są czytane/zapisywane
- Napisana jest w czystym C

6.1.1 Zastosowania w firmie Wooga

Rozwiązanie tego typu jest stosowane przez firmę Wooga [12], która migrowała część swoich gier z MySQL na Redis. Spowodowało to znaczne polepszenie wydajności serwera bazy danych, będącego wąskim gardłem dla całego systemu.

6.1.2 Podejście hybrydowe

W grze Monster World zastosowano rozwiązanie hybrydowe, to znaczy część danych pozostawiono w bazie MySQL, natomiast dane najczęściej zapisywane przeniesiono do Redis (największe obciążenie wywołane było właśnie przez zapis danych). W tej drugiej bazie przechowywane są dane dotyczące plecaka. Dzięki temu rozwiązaniu udało się w niespełna 2 tygodnie zmniejszyć obciążenie serwerów baz mySQL o 10%.

6.1.3 Całkowita migracja na Redis z wykorzystaniem pamięci wirtualnej

Z kolei w grze Happy Hospital baza danych została przeniesiona na Redis w całości. W celu zwiększenia przestrzeni w bazie danych zastosowano pamięć wirtualną (serwer posiadał 24GB ram + 380GB plik swapu). Dzięki temu rozwiązaniu aktualnie używane dane były przechowywane w szybkiej pamięci RAM. Osiągnięto bardzo dobrą responsywność serwera bazy danych nawet przy znacznym obciążeniu.

6.2 Inne zastosowania baz typu klucz-wartość

- Firma Zynga stosuje Membase w swoich produkcjach (między innymi w Farmville[13])
- OpenPoker wykorzystuje Mnesie [14]

6.3 Alternatywy wobec baz przechowujących dane w RAMie

Jeżeli ilość danych jaką trzeba przechowywać w bazie jest na tyle duża, że nie zmieszczą się one w RAMie lub pamięci wirtualnej, istnieją bazy typu klucz – wartość przechowujące dane na dysku. Przykładem takiej bazy jest Riak[15].

7. Dokumentowa baza noSQL

Tego typu bazy wydają się bardzo dobrze spełniać potrzeby gier komputerowych. Z jednej strony są bardzo szybkie, z drugiej oferują bardziej rozbudowany model danych, jakim jest dokument będący zbiorem wartości typu klucz-wartość zorganizowanym w drzewiastej strukturze. W dalszej części artykułu omówione zostaną dwa najpopularniejsze rozwiązania tego typu, czyli MongoDB i CouchDB.

7.1 MongoDB [16]

MongoDB, to baza zorientowana przede wszystkim na szybkość działania oraz skalowalność.

- Trójpoziomowa struktura: kolekcje przechowują klucze, które odpowiadają konkretnym dokumentom.

- Dokumenty przechowywane są na dysku w formacie BSON (binarny JSON), pozwalający na optymalizacje zapisu/odczytu. Ponadto silnik MongoDB prowadzi zapis w taki sposób, żeby był on możliwie szybki. Jest to realizowane przez updaty w miejscu, oraz cachowanie poleceń.

- Ograniczenie wielkości dokumentu do 16MB, do przechowywania większych porcji danych służy GridFS

- Jest bazą nastawioną na szybkość oraz skalowalność, w związku z tym nie obsługuje transakcji, gwarantuje za to atomowość na poziomie pojedynczych operacji.

- Pozwala na definiowanie indeksów na polach dokumentu

- Posiada wbudowany język zapytań bazujący na Javascript. Struktura zapytań jest intuicyjna dla osób mających styczność z SQL

- Pozwala na wykorzystanie mechanizmu MapReduce [17]
- Dwa sposoby replikacji, master – slave oraz replica sets
- Automatyczny sharding
- Napisana w C++, pierwszy release w listopadzie 2009

7.1.1 Event Sourcing jako alternatywa dla transakcji [18]

Aplikacja wykorzystująca wzorzec Event Sourcing zapisuje wszystkie zdarzenia, które wpływają na jej stan. Znany jest również stan początkowy aplikacji. Aktualny stan można odtworzyć przywracając stan początkowy i przetwarzając wszystkie zapisane zdarzenia. Takie podejście sprawia, że w przypadku niespodziewanej awarii, aplikacja jest w stanie wznowić działanie.

Zalety:

- Nie są potrzebne transakcje do zapewnienia spójności danych
- Wysoka wydajność, nie potrzeba żadnych locków
- Możliwość prześledzenia historii stanów aplikacji
- Snapshoty stanu aplikacji można wykonywać równolegle, nie zatrzymując jej działania

Wady:

- Oprogramowanie tego typu aplikacji jest zazwyczaj trudniejsze od bazującej na transakcjach.
- Rekonstrukcja stanu aplikacji na podstawie zdarzeń może być czasochłona.
- Trudność integracji z zewnętrznymi usługami. Przykładowo jeżeli aplikacja odpytuje zewnętrzny serwis, nie zawsze istnieje gwarancja, że odpowiedź na zapytanie będzie za każdym razem taka sama. Gdy odpowiedź wpływa na stan aplikacji, przestaje on być zależny tylko od zdarzeń, przez co nie da się stosować ES. Rozwiązaniem jest stosowanie tak zwanych bramek, które zapisują historie zapytań i odpowiedzi, dzięki czemu przy odtwarzaniu stanu możemy mieć

pewność, że wpływ zewnętrznego serwisu będzie zawsze taki sam. Nie zawsze jednak rozwiązanie to jest możliwe do zrealizowania w praktyce.

- Problematyczne wprowadzanie zmian w kodzie aplikacji. Jeżeli logika przetwarzania zdarzeń ulegnie zmianie, nie będzie możliwe odtworzenie aktualnego stanu na ich podstawie. Przy takiej zmianie zawsze trzeba zrobić snapshot aplikacji i logować zdarzenia od początku.

  Omawiany Wzorzec stosowany jest w grze ProjectArc[19].

7.1.2 Podejście hybrydowe

W niektórych aplikacjach da się oddzielić dane wrażliwe (np. związane z płatnościami) od innych (np. to w co postać jest aktualnie ubrana, zapis czatu itp.). W takiej sytuacji opłaca się zastosować bazę relacyjną do danych wrażliwych i MongoDB do pozostałych. W ten sposób nie ryzykujemy utraty spójności danych, zyskując większą szybkość działania.

Rozwiązanie to zastosowało Fuero Games w swojej grze MMO iDolina.pl. Serwer gry używa relacyjnej bazy Postgresql oraz MongoDB [20].

7.1.3 Atomowość na poziomie zapytania

MongoDB gwarantuje atomowość na poziomie pojedynczych zapytań. Przy odpowiedniej konfiguracji, metoda wywołująca zapytanie w kodzie aplikacji oczekuje na rezultat wywołanej operacji, i informuje, czy nie wystąpiły błędy.

Jeżeli zachowanie spójności danych nie wymaga przeprowadzenia niepodzielnej sekwencji updatów/insertów/deletów, transakcje nie są potrzebne.

7.1.4 Zastosowanie MongoDB w niedużym projekcie

WordSquared to krzyżówkowa gra MMO, której pierwsza wersja powstała podczas konkursu „Node Knockout”[21]. Zasady były proste: w ciągu 48 godzin trzeba stworzyć jak najlepszą grę, korzystającej z technologii node.js. Bazą danych wybraną przez twórców było właśnie MongoDB.

Krótko po ukończeniu gry okazało się, że cieszy się ona ogromnym zainteresowaniem, liczba aktywnych użytkowników rosła w lawinowym tempie. Miesiąc po premierze było ich sto tysięcy. Mimo tak zaskakujących dla twórców zdarzeń, MongoDB dzięki swojej skalowalności w pełni zdało egzamin i jest stosowane do dziś w tej produkcji.

Przykład tej gry dowodzi również, że bazy noSQL sprawdzają się nie

tylko w przypadku produkcji na ogromną skalę.

7.1.5 Indeksowanie przestrzenne

Indeksy przestrzenne pozwalają na wykonywanie szybkich zapytań na danych, będącymi koordynatami na powierzchni 2d lub na powierzchni kuli. Pozwalają one:

- Wylistować n dokumentów znajdujących się najbliżej od zadanego punktu
- Znaleźć wszystkie dokumenty których koordynaty znajdują się w zadanym okręgu lub prostokącie

Ta funkcjonalność została wykorzystana w omawianej wcześniej grze WordSquared. Gra ta podobna jest do popularnego Scrabble, ale toczy się na nieskończonej planszy. Do tej pory gracze ułożyli na niej miliony liter. Dzięki indeksowaniu przestrzennemu możliwe jest szybkie rysowanie widoków planszy.

Indeksowanie przestrzenne może być bardzo pomocne w bardzo wielu gatunkach gier. Mogą to być strategie podobne do OGame, MMORPGi, gry farmerskie. Generalnie każdy rodzaj gry, który zawiera dużą mapę, może wykorzystać tą funkcjonalność.

7.1.6 Inne zastosowania

- Disney stworzył repozytorium obiektów, na potrzeby produkowanych przez siebie gier, które jest oparte o MongoDB. (1400 instancji)
- Firma Metamoki zajmuje się tworzeniem gier społecznościowych między innymi na platformę Facebook. W jej produkcjach MongoDB jest szeroko stosowane.

7.2 CouchDB [22]

CouchDB ma strukturę podobną do MongoDB, obsługuje jednak transakcje

- W przeciwieństwie do MongoDB nie ma podziału na kolekcje. Klucz jest jedynym identyfikatorem dokumentu

- Dokumenty są zapisane tekstowo w JSONie.
- Przy zapisie danych stosowana jest strategia multiversion concurrency control, gwarantująca ACID.
- Nie posiada wbudowanego mechanizmu shardingu, pozwala na replikacje
- Możliwość definiowania widoków, które są cacheowane i są odpowiednikiem zapytań. Definiowanie widoków realizowane przy pomocy MapRediuce
- Brak indeksów zakładanych na pola dokumentu
- Napisany w Erlang, wersja 1.0 wypuszczona 10 lipca 2010

7.2.1 Multiversion concurrency control

Systemy bazodanowe implementujące MVCC nigdy nie nadpisują danych. Zamiast tego podczas np. updatu nowe dane są dopisywane i oznaczane identyfikatorem transakcji lub znacznikiem czasowym. Dane są wersjonowane.

Dzięki takiemu podejściu da się stosować transakcje na poziomie izolacji read uncommited [23] bez stosowania blokad, izolacja na wyższym poziomie wymaga ich stosowania.

Wadą takiego podejścia jest stale rosnące nadmierne wykorzystanie dysku, dlatego CouchDB cyklicznie usuwa stare zapisy.

7.2.2 Zastosowania

- Baza membase połączła się z projektem couchDB. Membase była szeroko stosowana przez Zynga, całkiem możliwe, że w przyszłości przejdzie na coucheDB [1].

8. Podsumowanie

NoSQL to naprawdę wspaniałe rozwiązanie dla gier online. Ich szybkość oraz prostota ułatwiają pracę przy małych projektach, takich jakim początkowo był WordSquared i wręcz umożliwiają tworzenie naprawdę

dużych. Gdyby firmy takie jak Zynga nie zaryzykowały odejścia od starych sprawdzonych baz relacyjnych, całkiem możliwe, że nie odniosły by takiego sukcesu. Prawdopodobnie byłoby też o wiele mniej gier dostępnych w modelu free to play.

Oczywiście relacyjne bazy danych mają swoje zastosowania, w których sprawdzają się bardzo dobrze, również w grach, lecz w wielu przypadkach można uniknąć ich użycia. Jeżeli nasz projekt potrzebuje mieć dostęp do ogromnej liczby niedużych porcji informacji, można stosować bazę podobną do Redis, gdy dane są cięższe, a szybkość jest sprawą kluczową, MongoDB jest idealnym rozwiązaniem, gdy ponadto potrzebujemy transakcji jest nim CouchDB. Czasami zdarza się, że tylko mały podzbiór danych, którymi operuje nasz projekt, wymaga użycia bazy relacyjnej, wtedy można stosować rozwiązania łączone. Możliwości jest naprawdę dużo.

Podsumowując: bazy relacyjne, to rozwiązania tworzone z myślą o klientach takich jak banki i tam sprawdzają się najlepiej. Przy tworzeniu gier istnieje bardzo dużo powodów żeby eksperymentować z nowymi technologiami takimi jak bazy noSQL.

Bibliografia

[1] Spis baz noSQL, http://nosql-database.org/
[2] Damian Skalski, „NoSQL – nierelacyjne systemy baz danych”, Software Developer’s Journal Sierpień 2011, s. 10-16
[3] Definicja shardingu, http://www.codefutures.com/database-sharding/
[4] Definicja failover z wikipedii, http://en.wikipedia.org/wiki/Failover
[5] Wpis dotyczący bazy Oracle na wikipedii, http://en.wikipedia.org/wiki/Oracle_Database
[6] Lista gier typu MMORPG które przeszły na model Free to Play w 2011 roku, http://mmorpg.org.pl/news/zobacz/Lista-MMORPG-kt%C3%B3re-w-2011-przesz%C5%82y-na-Free-mium
[7] World of Warcraft za darmo do 20 poziomu doświadczenia, http://hol-games.pl/wow-za-darmo-do-20-lvl.html
[8] Artykuł opisujący rozwiązania stosowane w firmie Playfish, http://highscalability.com/blog/2010/9/21/playfishs-social-gaming-architecture-50-million-monthly-user.html
[9] Wpis twórcy Minecrafta informujący o 20 milionach użytkowników, http://twitter.com/#!/notch/status/158311410600906753
[10] Strona domowa gry Gizarma, http://www.gizarma.pl
[11] Strona domowa bazy Redis, http://redis.io/
[12] Prezentacja firmy Wooga opisująca migrację ich projektów z bazy MySQL na Redis, http://www.slideshare.net/wooga/redis-to-the-rescue
[13] Artukuł opisujący użycie bazy Membase w grze Farmville, http://www.readwriteweb.com/cloud/2010/08/membase-the-database-powering.php
[14] Opis rozwiązań zastosowanych w grze Open Poker, http://pylab.blogspot.com/2007/03/openpoker-how-it-works.html
[15] Strona domowa gry bazy Riak, http://wiki.basho.com/

[16] Kyle Banker, „MongoDB in Action", 2011 Manning Publications
[17] Wpis na wikipedii dotyczący map-reduce, http://en.wikipedia.org/wiki/MapReduce
[18] Opis wzorca Event Sourcing, http://martinfowler.com/eaaDev/EventSourcing.html
[19] Urszula Trzaskowska, Michał Trzaskowski, „Wytwarzanie biznesowych aplikacji webowych a produkcja rozbudowanych gier na przeglądarkę na przykładzie Project Arc", *Wytwarzanie Gier Komputerowych*, s. 213-223, Gdańsk 2011
[20] Nagranie wykładu z konferencji WGK 2011 „iDolina.pl - kulisy produkcji 3D MMO w Unity3D", http://wgk2011.eti.pg.gda.pl/lecture/26/;jsessionid=7a5de3f3ef35daedac9afacd351a
[21] Strona na której opisano w jak została napisana gra WordSquared, http://www.startupmonkeys.com/2010/09/building-a-scrabble-mmo-in-48-hours/
[22] Strona domowa bazy CouchDB, http://couchdb.apache.org/
[23] Wpis na Wikipedii dotyczący izolacji w transakcjach, http://en.wikipedia.org/wiki/Isolation_%28database_systems%29

Use of noSQL databases in game development

Abstract

The paper describes the use of noSQL databases in online games, as an alternative to traditional relational databases. The characteristics are discussed of the databases most commonly used by the creators of games. On the examples of Redis, MongoDB and CouchDB advantages, disadvantages and solutions applied in existing productions are discussed. In addition, an analysis of requirements of computer games is performed. In conlcusion the rationale is presented indicating why game authors should use new database technologies.