|  |
| --- |
| Wojciech Ganobis Lista 7  Redaktor: |

1. NIe, ponieważ dostęp do każdego replikowanego obiektu jest szeregowany.
2. Używa się ich w celu zwielokrotnienia wydajności.
3. Zwielokrotnienie w systemach DNS polega na tym, że pamięć podręczna serwerów nazw wcześniej wyszukuje wuniki. Potem są one przechowywane w pamięci podręcznej, z dziaa to dobrze bo odwzorowanie tłumaczeń nazw na adresy nie znienia się często.
4. Jeśli program oczekuje sekwencyjnie spójnej pamięci danych i nie może działać z niczym innym, pamięć musi zapewniać sekwencyjną spójność. Jednakże, aby poprawić wydajność, niektóre systemy zapewniają słabsze modele. Konieczne jest zatem, aby oprogramowanie zgodziło się przestrzegać narzuconych zasad przez te modele.
5. A i B powinny wymienić swoją listę operacji wstępnych i posortować ją zgodnie z czasem.
6. Tak, ponieważ procesy przebiegają w kolejności a, c, b.
7. To zależy. Wielu programistów chroni swoje współdzielone dane poprzez mechanizmy synchronizacji. Głównym problemem jest to, że wymagają one grubszego ziarna niż te oferowane na poziomie odczytu i zapisu.
8. Tak. Czynnik powiązań końcowych stwierdza, że problemy należy rozwiązywać na tym samym poziomie, na którym występują. W tym przypadku mamy problem z całkowicie uporządkowanym rozsyłaniem w celu uzyskania spójności zwielokrotnienia. W protokołach bazowych jest spójność osiągana przez przekazanie wszystkich operacji do serwera głównego. Używając procesu porządkowania, robimy to samo ale na niższym poziomie abstrakcji.
9. Spójność przyczynowa. Reakcje na zmiany wartości giełdy powinny być spójne.
10. Najprostszą implementacją takiej skrzynki jest implementacja protokołu lokalnego zapisu, w którym podstawa zawsze znajduje się na komputerze użytkownika.
11. Najprostszą implementacją jest umożliwienie przeglądarce, aby zawsze sprawdzała, czy pokazuje najnowszą wersję strony. Wymaga to wysłania zapytania do serwera.
12. Nie stoi nic na przeszkodzie aby klient korzystał ze wspólnego planu. W takim przypadku aktualizacje mogą polegać na zaplanowaniu spotkania w czasie, w którym klient już zaplanował coś jeszcze, ale ta informacja nie została jeszcze przekazana innym repliką.
13. Nie.
14. Nie
16. (1,10), (2, 9), (3, 8), (4, 7), (5, 6), (6, 5), (7, 4), (8, 3), (9, 2) oraz (10, 1)
18. Tak
19. Tak. Szyja szereguje żądania, aby pojawiały się w pamięci w porządku czasowym.
20. Ponieważ narusza to spójność danych.
21. Nabywanie jest potrzebne, aby opóźnić proces próbujący uzyskać dostęp do zmiennych współużytkowanych, gdy robi to inny proces.
22. Orca zapewnia tylko spójność wpisów, ponieważ współbieżne operacje na tym samym obiekcie są odpowiednio serializowane. Jednak w przypadku korzystania z całkowicie uporządkowanego nadawania jako środka do realizacji porządkowania operacji wszystkie operacje są uporządkowane globalnie, niezależnie od obiektu, na którym są wykonywane. W takim przypadku oferuje sekwencyjną spójność.
23. Jeśli nie mamy założenia, że każdy element danych ma jednego właściciela. Na przykład, jeśli dwóch niezależnych użytkowników tych samych danych ostatecznie wiąże się z tą samą repliką, ich odpowiednie aktualizacje mogą wymagać propagacji do tej repliki.
24. Załóżmy, że obiekt A wywołuje B. Każdemu wywołaniu jest przypisywany ten sam identyfikator przez wszystkie repliki A. Następnie każda replika zwielokrotnia swoje żądanie wywołania do replik B. Gdy replika B otrzymuje żądanie wywołania, sprawdza, czy otrzymała już to żądanie z jednej z replik A. Jeśli nie, wywołanie jest wykonywane w lokalnej replice, a komunikat odpowiedzi jest rozsyłany grupowo do wszystkich replik A. Jeśli wywołanie zostało już wykonane, przychodzące żądanie jest dalej ignorowane. Podobnie replika A przekazuje tylko pierwszą odpowiedź przychodzącą dla zaległego żądania wywołania do swojej lokalnej kopii obiektu, podczas gdy wszystkie kolejne odpowiedzi na to samo żądanie wywołania są po prostu ignorowane.
25. Jeśli wiadomo że dana operacja została wykonana wszędzie.