Jedrzej Byrski, Pawel Slowikowsk (ps@agh.edu.pl)
Akademia Górniczo-Hutnicza, Kraków, Polska

SERWIS TOLEROWANIA USZKODZEN DLA APLIKACJI OPARTYCH NA ARCHITEKTURZE CORBA

Streszczenie: *Wspólczesne systemy komputerowe staja sie coraz bardziej skomplikowane, a projektanci oprogramowania coraz czesciej siegaja po rozproszone architektury obiektowe. Aby zapewnic komfort tworzenia takich aplikacji oraz skrócic czas ich powstawania zaprojektowany zostal serwis tolerowania uszkodzen. Dostarcza on mechanizmy obslugujace awarie systemu, przez co uwalnia projektantów od zmudnego i czasochlonnego opracowywania ich od podstaw.*

Slowa kluczowe: CORBA, fault tolerance, middleware, object groups, replication, checkpointing, recovery
tolerowanie uszkodzen, replikacja, punkty kontrolne, odtwarzanie stanu.

1 Wstep

Ostatnio, coraz bardziej popularne staje sie wykorzystywanie rozproszonych architektur obiektowych do budowy zlozonych rozproszonych systemów. Przykladami takich architektur zyskujacych obecnie duza popularnosc sa CORBA, RMI i DCOM. Warstwa middleware wprowadzona przez te architektury pozwala rozproszonym obiektom na komunikowanie sie w modelu klient-serwer. Klienci maja dostep do uslug oferowanych przez obiekty serwerów za pomoca zdalnych wywolan metod. Kazdy obiekt serwera posiada zdefiniowany interfejs okreslajacy zbiór metod które moga byc zdalnie wywolane przez klienta. Za wszystkie szczególy zwiazane ze zdalnym wywolaniem metody (m.in. marshaling i unmarshaling argumentów i rezultatów metod) odpowiedzialna jest warstwa middleware.

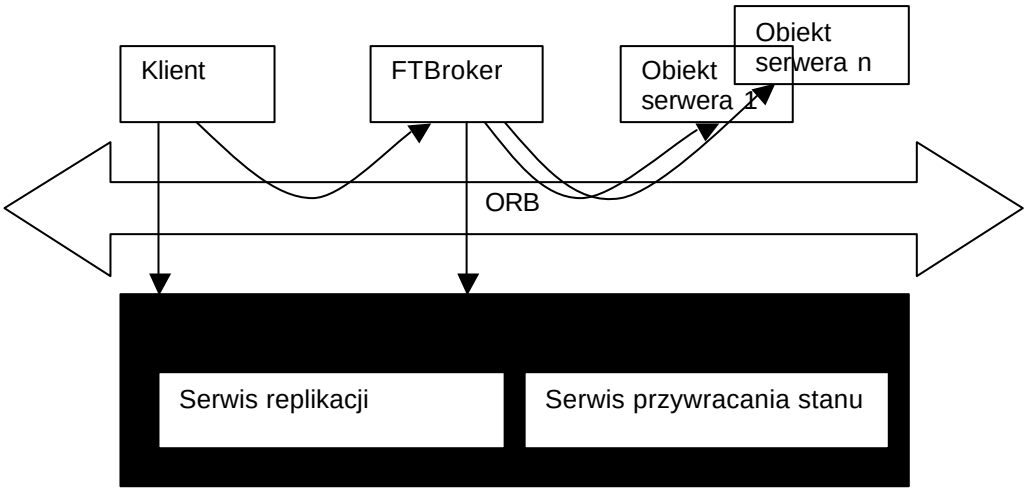
Wymienione powyzej rozproszone architektury obiektowe skupiaja sie glównie na zapewnieniu przenosnosci, wspóldzialania i wielokrotnego uzycia stworzonych obiektów i komponentów. Niestety, zadna z nich nie pozwala na tworzenie aplikacji w pelni niezawodnych i odpornych na uszkodzenia. Stanowi to, w przypadku wielu zlozonych i z zalozenia niezawodnych aplikacji, powazny brak który pozostaje do wypelnienia samemu. Zabezpieczenie sie jednak przed wszystkimi potencjalnymi mozliwosciami awarii systemu jest trudne, czasochlonne i czesto powoduje powstawanie nowych uszkodzen.

Rozwiazanie tego problemu zapewnilby serwis tolerowania uszkodzen uodparniajacy aplikacje na mogace wystapic w czasie jej dzialania uszkodzenia. Korzystanie z takiego serwisu powinno ograniczac do minimum zmiany wymagajace wprowadzenia do istniejacej aplikacji w celu stworzenia z niej aplikacji w pelni odpornej na uszkodzenia.

Celem tej pracy jest przedstawienie projektu architektury serwisu tolerowania uszkodzen. Publikacja ma nastepujaca strukture: rozdzial 2. przedstawia budowe serwisu tolerowania uszkodzen oraz opisuje poszczególne jego skladniki, rozdzial 3. skupia sie na metodach przywracania stanu systemu sprzed wystapienia uszkodzenia, rozdzial 4. omawia zastosowane metody replikacji danych i rozdzial 5. prezentuje koncowe wnioski.

2 Budowa serwisu tolerowania uszkodzen

Serwis tolerowania uszkodzen sklada sie z serwisu replikacji (*replication service*) i serwisu przywracania stanu (*recovery service*). Mechanizm tolerowania uszkodzen dostarczonych przez te serwisy opiera sie na komunikacji grupowej, punktach kontrolnych(*checkpointing*) oraz replikacji.



Rys. 0. Uproszczona budowa serwisu tolerowania uszkodzen

2.1 Serwis replikacji

Serwis replikacji odpowiedzialny jest za tworzenie oraz przechowywanie definicji grup, czyli kolekcji rozproszonych obiektów serwerów o identycznej funkcjonalnosci. W przypadku zadania aktywowania grupy tworzony jest FTBroker reprezentujacy grupe, a jego referencja zwracana jest do klienta.

2.2 FTBroker

FTBroker posredniczy w komunikacji pomiedzy klientem a obiektami serwerów. Pelni on wiele kluczowych dla serwisu funkcji:

- przywraca obiekty serwerów po ich uszkodzeniu do stanu sprzed awarii,
- dzieki wprowadzeniu redundantnosci obiektów serwerów uodparnia zarówno na uszkodzenia powstale w czasie dzialania poszczególnych obiektów jak i na bledy podczas komunikacji z tymi obiektami,
- zapamietuje konsystentne stany oraz operacje wykonane na obiektach serwerów. Dzieki tym informacja mozliwe jest przywracanie stanu obiektów po awarii systemu,
- zapewnia spójnosc stanów obiektów serwerów poprzez wykorzystanie mechanizmów synchronizacji i replikacji,
- chroni przed wielokrotnym wykonaniem tej samej operacji na poszczególnych obiektach serwerów.

2.3 Serwis przywracania stanu

Glównym zadaniem serwisu przywracania stanu jest przechowywanie informacji o stanach obiektów serwerów oraz wykonanych na nich operacjach. Oprócz nazw operacji pamietane sa takze argumenty operacji oraz zwracane rezultaty. FTBroker w czasie przywracania dzialania obiektów serwerów wykorzystuje serwis przywracania stanu do pobierania informacji niezbednych do uaktualnienia ich stanów.

2.4 Klient

Klient widzi obiekt serwera poprzez FTBrokera, tzn. posiada zamiast referencji do obiektu serwera referencje do reprezentujacego go FTBrokera. Wszystkie operacje, które wykonywalby bezposrednio obiekt serwera, przekazywane sa do FTBrokera. Nastepnie FTBroker po otrzymaniu rezultatu od odpowiedniego obiektu serwera zwraca rezultat do klienta. W ten sposób klient calkowicie izolowany jest od obiektów serwerów.

2.5 Obiekt serwera

Obiekt serwera wspólpracujacy z FTBrokerem jest zwyklym obiektem wyposazonym dodatkowo w odpowiedni interfejs. Dwie kluczowe metody które musi dostarczac obiekt serwera to `GetState()` oraz `SetState()`. Pierwsza z tych metod zwraca stan obiektu, czyli informacje mogace posluzyc do odtworzenia stanu w jakim obiekt sie znajduje lub informacje, ze stan obiektu jest niekonsystentny i nie powinien byc zachowany. Druga z metod przywraca obiekt do stanu otrzymanego jako rezultat pierwszej metody. Obiekt serwera wyposazony jest w mechanizm zapobiegajacy wielokrotnemu wykonaniu tej samej operacji.

3 Przywracanie stanu aplikacji sprzed wystapienia bledu

W czasie dzialania systemu moze dojsc do róznych awarii. Aplikacje wykorzystujace serwis tolerowania uszkodzen powinny po zajsciu awarii zostac przywrócone do stanu pelnej funkcjonalnosci. Czesto jednak podczas awarii systemu oprócz uszkodzenia aplikacji zmienia sie bezpowrotnie stan srodowiska w którym aplikacja funkcjonuje. W takiej sytuacji aplikacja powinna dynamicznie dostosowac sie do nowych warunków dzialania. Ponizej przedstawione sa sposoby przywracania dzialania róznych elementów systemu po ich uszkodzeniach. Zalozone zostalo, ze awarie nie uszkadzaja serwisu tolerowania uszkodzen i dziala on caly czas poprawnie.

3.1 Przywracanie dzialania FTBrokera

Klient, w przypadku niemoznosci wykonania operacji na FTBrokerze, kontaktuje sie z serwisem replikacji i prosi o stworzenie nowego FTBrokera. Serwis replikacji tworzy FTBrokera. FTBroker kontaktuje sie z wszystkimi obiektami serwerów nalezacymi do grupy i pobiera informacje o ich stanach. Na podstawie danych uzyskanych od serwisu przywracania stanu FTBroker aktualizuje stany obiektów serwerów wymagajacych aktualizacji.

3.2 Przywracanie dzialania i resynchronizacja obiektu serwera

Po uszkodzeniu obiektu serwera tworzony jest nowy obiekt. FTBroker na podstawie danych uzyskanych od serwisu przywracania stanu aktualizuje stan obiektu. Proces ten, w uproszczeniu, polega na przywróceniu obiektowi serwera ostatniego konsystentnego stanu, a nastepnie wykonaniu wszystkich operacji wywolanych przez klienta po zapisaniu konsystentnego stanu.

3.3 Przywracanie dzialania FTBrokera i obiektu serwera

Przywracanie dzialania FTBrokera i obiektu serwera po równoczesnej awarii jest podobne do przywracania dzialania samego FTBrokera. Rozszerzeniem jest o to, ze nowo stworzony przez serwis replikacji FTBroker musi dodatkowo przywrócic dzialanie i zresynchronizowac obiekt serwera.

4 Mechanizm replikacji

Mechanizm replikacji zostal wprowadzony w celu zapewnienia wiekszej wydajnosci przywracania systemu do dzialania po uszkodzeniu. W przypadku niepoprawnego dzialania jednego z obiektów serwerów obsluge przychodzacych zadan moze natychmiast przejac inna jego replika minimalizujac narzuty czasowe wprowadzone przez wystapienie uszkodzenia.

Za zarzadzanie replikacja odpowiedzialny jest FTBroker. Gwarantuje on, ze po otrzymaniu zadania wszystkie obiekty serwerów tworzace grupe osiagna konsystentne stany. Zastosowane zostaly trzy typy replikacji: pasywna, aktywna oraz hybrydowa (bedaca kombinacja dwóch wczesniejszych). Kazda z tych replikacji moze wystepowac w dwóch wersjach:

- stany wszystkich obiektów grupy sa uaktualnione przed zwróceniem do klienta rezultatu zadania,
- rezultat zadania zwracany jest do klienta zaraz po otrzymaniu go przez FTBrokera od jednego z obiektów serwerów. Stany pozostalych obiektów serwerów aktualizowane sa w nastepnej kolejnosci.

Ponizej przedstawione sa zastosowane modele replikacji w wersji pierwszej.

4.1 Replikacja pasywna

1. Klient wywoluje operacje na FTBrokerze.
2. FTBroker wywoluje operacje na pierwszym z obiektów serwerów.
3. Obiekt serwera wykonuje zadana operacje, w wyniku czego zmienic sie moze jego stan. Nastepnie zwraca do FTBrokera rezultat operacji wraz z nowym stanem.
4. FTBroker przesyla stan otrzymany od pierwszego serwera do pozostalych obiektów w grupie.
5. Obiekty aktualizuja swoje stany do nowo otrzymanego.

6. Po zakonczeniu aktualizacji FTBroker zwraca rezultat operacji do klienta.

4.2 Replikacja aktywna

1. Klient wywoluje operacje na FTBrokerze.
2. FTBroker wywoluje operacje na wszystkich obiektach serwerów w w grupie.
3. Kazdy obiekt serwera wykonuje operacje i zwraca rezultat do FTBrokera.
4. FTBroker zwraca do klienta pierwszy otrzymany rezultat.

4.3 Replikacja hybrydowa

Czesc obiektów replikowana jest pasywnie, czesc aktywnie.

5 Wnioski

Przedstawione rozwiazanie nastawione jest na wykorzystanie w architekturze CORBA, nic jednak nie stoi na przeszkodzie aby mozna je bylo uzyc w dowolnej innej rozproszonej architekturze. Na szczególna uwage zasluguje mozliwosc zaprojektowania aplikacji tolerujacej uszkodzenia w taki sposób, by mogla dostosowywac sie do nieodwracalnych zmian srodowiska w którym dziala wywolanych awaria systemu. Daje to mozliwosc wykorzystania serwisu tolerowania uszkodzen w szerokim zakresie systemów czasu rzeczywistego. Planowane jest wzbogacenie istniejacego serwisu o moduly monitorujace dzialanie obiektów serwerów oraz rozszerzenie protokolu aktualizacji stanów pozwalajace na zautomatyzowanie wymaganego implementowania interfejsu obiektu serwera.

Opisany w tej publikacji serwis tolerowania uszkodzen zostal zaimplementowany przez studentów Wydzialu Informatyki, Akademii Górniczo-Hutniczej w ramach pracy magisterskiej. Przy jego uzyciu stworzona zostala aplikacja w pelni tolerujaca uszkodzenia symulujaca system telefoniczny i rejestrujaca czasy rozmów.

Literatura:

[1] A. Avizienis, *Toward Systematic Design of Fault-Tolerant Systems*, IEEE, 1997.

[2] E. N. Elnozahy, D. B. Johnson, Y. M. Wang, *A Survey of Rollback-Recovery Protocols in Message-Passing Systems*, 1996.

[3] P. Felber, X. Défago, P. Eugster, A. Schiper, *Replicating CORBA Objects: A Marriage Between Active and Passive Replication*, IFIP TC6 WG6.1 Second International Working Conference on Distributed Applications and Intreoperable Systems (DAIS'99), Helsinki, Finland, 1999.

[4] R. Guerraoui, A. Schiper, *Software-Based Replication for Fault Tolerance*, IEEE, 1997.

[5] K. Guo, Luís Rodrigues, *Dynamic Light-Weight Groups*, Technical Report, TR96-1612, 1996.

[6] S. Maffeis, Piranha: *A CORBA Tool For High Availability*, IEEE, 1997.

[7] A. Montresor, *Fault Tolerance through View Synchrony in Partitionable Asynchronous Distributed Systems*, Technical Report UBLCS-96-16, 1997.

[8] A. Montresor, *The JGroup Distributed Object Model*, IFIP TC6 WG6.1 Second International Working Conference on Distributed Applications and Intreoperable Systems (DAIS'99), Helsinki, Finland, 1999.

[9] A. P. A. van Moorsel, *Action Models: A Reliability Modeling Formalism for Fault-Tolerant Distributed Computing Systems*, 1998.

[10] G. Morgan, S. Shrivastava, P. Ezhilchelvan, M. Little, *Design and Implementation of a CORBA Fault-Tolerant Object Group Service*, IFIP TC6 WG6.1 Second International Working Conference on Distributed Applications and Intreoperable Systems (DAIS'99), Helsinki, Finland, 1999.

[11] P. Pietras, P. Slowikowski, *Mostek bezpieczenstwa dla protokolu internet inter-orb protocol*. Zeszyty naukowe AGH. Informatyka. Tom 1. 1999.

[12] J. Schönwälder, S. Grag, Y. Huang, A. P. A. van Moorsel, S. Yajnik, *A Management Interface for Distributed Fault Tolerance CORBA Services*. Proc. 3rd IEEE International Workshop on Systems Management, Newport, RI, USA, 1998.

[13] A. S. Tanenbaum, *Rozproszone systemy operacyjne*, Wydawnictwo Naukowe PWN, 1997.

[14] J. Weissenfels, D. Wodtke, G. Weikum, A. Kotz Dittrich, *The Mentor Architecture for Enterprise-wide Workflow Management*, 1996.