Grażyna SUCHACKA
POLITECHNIKA OPOLSKA, WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI I INFORMATYKI, INSTYTUT AUTOMATYKI I INFORMATYKI
ul. Sosnkowskiego 31, 45-272 Opole

Biznesowo-zorientowana metoda obsługi żądań w serwisie WWW

Dr inż. Grażyna SUCHACKA

Ukończyła studia na kierunku Informatyka, specjalność Systemy Informacyjne, na Wydziale Informatyki i Zarządzania Politechniki Wrocławskiej. Na tym wydziale uzyskała również stopień doktora nauk technicznych w dyscyplinie informatyka. Pracuje w Katedrze Informatyki na Wydziale Elektrotechniki, Automatyki i Informatyki Politechniki Opolskiej. Zajmuje się zagadnieniami związanymi z jakością usług serwisów WWW, ze szczególnym uwzględnieniem aplikacji handlu elektronicznego.

e-mail: g.suchacka@po.opole.pl

Streszczenie

Artykuł dotyczy problemu gwarantowania jakości usług serwisów WWW zapewniających funkcjonowanie detalicznych sklepów internetowych. Zaproponowano nową metodę obsługi żądań w serwisie, której celem jest maksymalizowanie przychodu osiąganego przez właściciela e-biznesu, przy jednoczesnym oferowaniu wyższej jakości usług dla bardziej wartościowych klientów. Do identyfikacji i oceny wartości kluczowych klientów zaproponowano zastosowanie analizy RFM (ang. *Recency-Frequency-Monetary value* analysis). Przedyskutowano nowy algorytm kontroli przyjęć i szeregowania żądań, realizujący sterowanie zgodnie z przyjętymi celami „biznesowymi".

Słowa kluczowe: serwis WWW, kontrola przyjęć, szeregowanie, handel elektroniczny, B2C, jakość usług, QoWS.

Business-oriented request service method for a Web server system

Abstract

The paper deals with the problem of guaranteeing high Quality of Web Service (QoWS) in e-commerce Web servers. Due to the high variability and unpredictability of Web traffic, Web servers are subject to overloads, which result in users experiencing too long response times, their impatience and site abandonment. Such situations are detrimental to e-business conducted over the Internet, especially in the highly competitive Business-to-Consumer (B2C) e-commerce environment. In the paper, a novel request service method for a Web server system hosting a B2C Web site is proposed. The method aims at ensuring high revenue achieved by an online-retailer through successfully processed purchase transactions, as well as offering higher QoWS for more valued customers. To identify and evaluate values of key customers, RFM (*Recency-Frequency-Monetary value*) analysis has been applied to the method. A new admission control and scheduling algorithm realizing request service control according to business-oriented goals is discussed.

Keywords: Web server, admission control, scheduling, e-commerce, B2C, quality of Web service, QoWS.

1. Wprowadzenie

W ostatnich latach można zaobserwować rosnące zainteresowanie usługami realizowanymi za pośrednictwem WWW (ang. *World Wide Web*). Web stanowi szczególnie obiecującą platformę dla handlu elektronicznego, pozwalając użytkownikom Internetu na wyszukiwanie informacji o produktach, porównywanie cen i dokonywanie zakupów za pośrednictwem globalnej sieci.

Sklepy internetowe są implementowane poprzez witryny B2C (ang. *Business-to-Consumer*). Witryna B2C składa się z wielu stron webowych, poprzez które użytkownik realizuje pewne typowe operacje „biznesowe", takie jak wyszukiwanie produktów, dodawanie ich do wirtualnego koszyka zakupów, itp. Podczas pojedynczej wizyty w sklepie internetowym użytkownik odwiedza zwykle wiele stron, czyli realizuje sekwencję operacji biznesowych, która składa się na *sesję użytkownika*.

Podstawą sprawnego funkcjonowania sklepu internetowego jest serwis WWW obsługujący witrynę B2C (w artykule taki serwis nazywany jest *biznesowym* serwisem WWW). Serwis składa się zwykle z jednego lub wielu serwerów webowych, wspomaganych przez serwery zaplecza. Jego zadaniem jest obsługa żądań HTTP nadchodzących przez Internet od klientów (ang. *clients*). Dla każdej strony webowej żądanej przez użytkownika, przeglądarka internetowa klienta generuje wiele żądań HTTP i wysyła je do odpowiedniego serwisu webowego. Pierwsze żądanie HTTP dla strony dotyczy pobrania dokumentu HTML zawierającego szkielet strony, natomiast kolejne żądania HTTP dotyczą pobrania obiektów zagnieżdżonych na stronie. Zwykle żądania nadchodzące do serwisu od różnych klientów obsługiwane są niezależnie od siebie, zgodnie z regulaminem obsługi FIFO (ang. *First-In-First-Out*).

W związku z ograniczonymi zasobami sprzętowymi każdego serwisu, a także zmiennym i nieprzewidywalnym charakterem ruchu webowego, powstaje problem jakości usług serwisów WWW (ang. *Quality of Web Service*, w skrócie QoWS). Wiąże się on z występowaniem przeciążeń serwisów, podczas których czasy odpowiedzi serwisu dla żądań HTTP znacznie rosną; z perspektywy użytkowników problem przejawia się długim czasem oczekiwania na wyświetlenie strony w przeglądarce lub brakiem odpowiedzi. Niska jakość usług negatywnie wpływa na e-biznes, m.in. na zaufanie użytkowników do bezpieczeństwa transakcji na witrynie, postrzeganie oferowanych produktów i wizerunek firmy. Negatywne skutki w dłuższej perspektywie to mniejsza liczba klientów i niższe przychody właściciela e-biznesu.

Jednym ze sposobów podniesienia QoWS jest zastąpienie szeregowania FIFO algorytmem kontroli przyjęć i szeregowania żądań zgodnie z pewnym przyjętym kryterium jakości usług. Wprowadzenie kontroli przyjęć umożliwia zapobieganie przeciążeniom serwisu poprzez odrzucanie niektórych żądań na jego wejściu, kiedy obciążenie przekracza pewien poziom. Z drugiej strony, szeregowanie żądań umożliwia zmianę kolejności ich obsługi w systemie tak, aby zapewnić krótsze czasy obsługi żądań ważniejszych z punktu widzenia przyjętego kryterium.

2. Biznesowo-zorientowane cele sterowania obsługą żądań

Przyjmując kryterium jakości usług dla biznesowego serwisu WWW należy uwzględnić specyfikę sesji użytkownika na witrynie B2C i jej aspekt finansowy. Czynniki te wskazują na potrzebę rozpatrywania działania serwisu nie tylko w kontekście jego wydajności jako pewnego rozwiązania systemu komputerowego, ale również z perspektywy rentowności e-biznesu. Charakterystyka obciążenia serwisów i sposobu nawigacji użytkowników na witrynach B2C, a także badania w dziedzinie zarządzania relacjami z klientem, wskazują na celowość zaproponowania metody zróżnicowanej obsługi żądań w serwisie WWW, która umożliwi oferowanie wyższej jakości usług tym użytkownikom, których można uznać za bardziej wartościowych klientów (ang. *customers*). Jednocześnie istotne jest zapewnienie jak największego przychodu dla właściciela e-biznesu. Zaproponowany w artykule sposób obsługi żądań uwzględnia oba te aspekty.

W dalszej części artykułu „klient" oznacza użytkownika (ang. *customer*), w przeciwieństwie do maszyny klienckiej (ang. *client*).

Jakość usług serwisu zdefiniowana została z perspektywy dwóch celów. Pierwszy cel dotyczy zapewnienia jak największego przychodu osiągniętego w wyniku pomyślnie obsłużonych transakcji zakupu poprzez witrynę B2C. Drugi cel dotyczy zapewnienia krótszych czasów odpowiedzi dla bardziej wartościowych klientów. W tym celu zdefiniowano klasę *kluczowych klientów*, obejmującą użytkowników, którzy w przeszłości dokonali zaku-

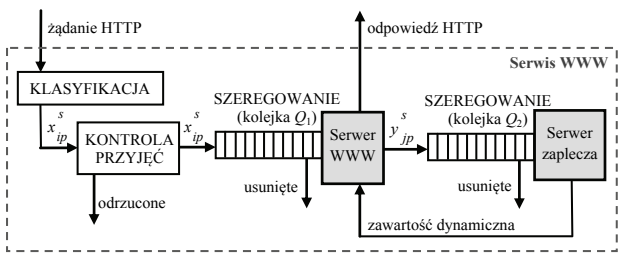
pów poprzez daną witrynę (w przeciwieństwie do *zwykłych klientów*, czyli użytkowników bez żadnego wcześniejszego zakupu).

Do obliczania wartości kluczowych klientów zaproponowano zastosowanie analizy RFM (ang. *Recency-Frequency-Monetary value*), która umożliwia obliczenie tzw. *wskaźnika RFM* klienta na podstawie historii jego zakupów. Zaproponowano utworzenie bazy danych kluczowych klientów w serwerze WWW, a następnie okresowe przeprowadzanie segmentacji bazy zgodnie z analizą RFM. Sposób jej przeprowadzania opisano w pracy [1].

Ze względu na cele sterowania, zaproponowana metoda obsługi żądań została nazwana KARO (*Key customers And Revenue Oriented admission and scheduling*).

3. Model sterowania obsługą żądań w biznesowym serwisie WWW – metoda KARO

Rozpatrywany jest biznesowy serwis webowy w architekturze wielowarstwowej, na którą składa się pojedynczy serwer webowy oraz pojedynczy serwer zaplecza (rys. 1). Wyróżniono dwa rodzaje żądań w serwisie: żądania HTTP i żądania dynamiczne.



Rys. 1. Model obsługi żądań w biznesowym serwisie WWW
Fig. 1. Model of request service in a business Web Server system

Żądania HTTP nadchodzą do serwisu od klientów przez Internet. Każde żądanie HTTP jest indeksowane kolejnym numerem *i*, numerem sesji *s* oraz numerem strony *p* w sesji *s* i jest oznaczone przez x^s_{ip}. Kiedy system jest mocno obciążony, niektóre żądania HTTP mogą być odrzucane na jego wejściu; przyjęte żądania HTTP zostają umieszczone w odpowiednio uszeregowanej kolejce Q_1 na wejściu serwera WWW. Żądanie może zostać od razu pobrane do obsługi albo oczekuje na pobranie przez pewien okres czasu, nie dłuższy niż T_1, po którym zostaje usunięte z kolejki.

Żądania dynamiczne są generowane i wysyłane przez serwer webowy do serwera zaplecza, jeżeli utworzenie odpowiedzi na żądanie HTTP wymaga wykonania bieżących obliczeń i dostępu do bazy danych. Każde żądanie dynamiczne przetwarzane jest więc w ramach obsługi odpowiedniego żądania HTTP i jest indeksowane kolejnym numerem *j* ($j \le i$), numerem sesji *s* oraz numerem strony *p* w sesji *s* i jest oznaczone przez y^s_{jp}. Żądania dynamiczne są umieszczane w odpowiednio uszeregowanej kolejce Q_2 na wejściu serwera zaplecza. Maksymalny czas oczekiwania w kolejce Q_2, po którym żądanie jest usuwane, oznaczono przez T_2.

Sterowanie obsługą żądań w serwisie można opisać jako wieloetapowy proces decyzyjny [2]. Niech *n* oznacza moment nadejścia żądania HTTP do serwera WWW albo moment nadejścia żądania dynamicznego do serwera zaplecza i niech będzie to moment wyznaczenia decyzji sterującej. Decyzje sterujące wyznaczane w chwilach *n* dotyczą przyjmowania bądź odrzucania żądań HTTP, a także wyznaczania uszeregowania żądań w kolejkach Q_1 i Q_2.

Ponadto, niech *m* oznacza moment ukończenia obsługi żądania HTTP (czyli pomyślnego przetworzenia żądania HTTP, odrzucenia żądania na wejściu systemu albo usunięcia go z kolejki).

W każdej chwili *n* z usług serwisu webowego korzysta jednocześnie określona liczba użytkowników, czyli możemy mówić o pewnej liczbie $S(n)$ sesji aktywnych i obsługiwanych przez system. W zależności od bieżącego obciążenia serwisu każda sesja może zostać ostatecznie pomyślnie ukończona albo przerwana.

Def. 1. Sesja *s* jest *pomyślnie ukończona* w chwili *m*, jeżeli żądanie x^s_{ip}, którego obsługa została ukończona w chwili *m*, było ostatnim żądaniem sesji *s* i czas odpowiedzi serwisu dla strony *p* nie przekroczył progu tolerancji opóźnienia przez użytkownika T_u.

Def. 2. Sesja *s* jest *przerwana* w chwili *m*, jeżeli: (1) żądanie należące do strony *p* w sesji *s* zostało odrzucone na wejściu systemu lub usunięte z kolejki Q_1 lub Q_2 w chwili *m* albo (2) jeżeli po przetworzeniu żądania x^s_{ip} w chwili *m* czas odpowiedzi serwisu dla strony *p* przekracza próg T_u.

Aby wyróżnić sesje najważniejsze z punktu widzenia biznesowo-zorientowanych celów sterowania, zaproponowano, aby każda sesja *s*, $s = 1, 2, ..., S(n)$, była w chwili *n* charakteryzowana przez następujące *atrybuty*:

- klasa sesji $c^s(n) \in \{KC, OC\}$, gdzie *KC* oznacza sesję kluczowego klienta, a *OC* sesję zwykłego klienta;
- ranga sesji $RFM^s(n)$, odzwierciedlająca wartość klienta w sesji *s*; dla klasy *KC* odpowiada ona wskaźnikowi *RFM* klienta odczytanemu z bazy danych, a dla klasy *OC* jest równa 0;
- stan sesji $e^s(n) \in \{H, L, B, S, D, A, R, P\}$, odpowiadający typowi bieżącej operacji biznesowej w sesji: wejściu na stronę główną (*H*), logowaniu (*L*), przeglądaniu (*B*), wyszukiwaniu produktów (*S*), opisowi produktu (*D*), dodaniu produktu do koszyka (*A*), rejestracji (*R*) albo dokonaniu zapłaty (*P*);
- długość sesji $l^s(n) = 1, 2, ...$, oznaczająca liczbę dotychczasowych stron webowych w sesji *s* włącznie z bieżącą stroną;
- wartość koszyka zakupów $v^s(n)$, odpowiadająca łącznej wartości finansowej produktów w koszyku zakupów sesji *s*.

W celu umożliwienia zróżnicowanej obsługi w serwisie żądań należących do różnych sesji, wprowadzono *priorytety* sesji, obliczane na podstawie atrybutów sesji z uwzględnieniem celów biznesowych. Priorytet sesji jest aktualizowany dla każdego żądania HTTP dotyczącego obiektu HTML, czyli dla nowej strony.

Niech T_{Med} i T_{Low} ($T_{Med} < T_{Low}$) oznaczają dwie wartości progowe, wyznaczające trzy przedziały dla długości sesji. Priorytet sesji *s* w chwili *n* obliczany jest następująco:

$$P^s(n) = \begin{cases} 4 \text{ dla } (c^s(n) = KC) \vee [(c^s(n) = OC) \\ \qquad \wedge (v^s(n) > 0) \wedge (e^s(n) = P)], \\ 3 \text{ dla } (c^s(n) = OC) \wedge [((v^s(n) > 0) \\ \qquad \wedge (e^s(n) \neq P)) \vee ((v^s(n) = 0) \\ \qquad \wedge (l^s(n) < T_{Med}))], \\ 2 \text{ dla } (c^s(n) = OC) \wedge (v^s(n) = 0) \\ \qquad \wedge (T_{Med} \le l^s(n) < T_{Low}), \\ 1 \text{ dla } (c^s(n) = OC) \wedge (v^s(n) = 0) \\ \qquad \wedge (l^s(n) \ge T_{Low}). \end{cases} \quad (1)$$

Priorytet 4 jest przyznawany wszystkim kluczowym klientom, jak również zwykłym klientom finalizującym transakcje zakupu. Priorytet 3 jest przyznawany wszystkim klientom w początkowym okresie interakcji z witryną, a także zwykłym klientom z niepustym koszykiem zakupów bez względu na długość sesji. Priorytety 2 i 1 są przyznawane klientom przebywającym na witrynie przez długi okres czasu z pustym koszykiem zakupów.

4. Algorytm KARO-Rev

Dla opracowanej metody KARO można zaproponować różne algorytmy kontroli przyjęć i szeregowania żądań. Poniżej przedstawiony jest heurystyczny algorytm KARO-Rev (*KARO – Revenue-oriented*), zorientowany na maksymalizowanie przychodu przy dodatkowym kryterium związanym ze wskaźnikami RFM.

Kontrola przyjęć realizowana jest następująco. Niech I_1 oraz I_2 ($I_1 < I_2$) będą pewnymi wartości progowymi dla obciążenia danego serwisu. Kiedy obciążenie przekroczy próg I_1, odrzucane są żądania HTTP dotyczące nowych stron webowych w sesjach o priorytecie 1. Powyżej progu I_2 odrzucane są żądania nowych stron w sesjach o priorytecie 1 i 2.

Szeregowanie żądań w kolejkach Q_1 i Q_2 umożliwia zmianę kolejności obsługi żądań odpowiednio w serwerze webowym i serwerze zaplecza. Żądania w każdej kolejce szeregowane są według polityki ścisłego priorytetu, tzn. wszystkie żądania o wyższym priorytecie mają pierwszeństwo obsługi przed żądaniami o niższym priorytecie. Ponadto, żądania należące do sesji o priorytecie 4 uporządkowane są względem siebie malejąco według wartości koszyka zakupów, przy czym żądania o tych samych wartościach koszyka dodatkowo są uporządkowane malejąco według rang sesji. Żądania należące do sesji o priorytecie 3 są uporządkowane malejąco według wartości koszyka zakupów, a żądania w sesjach o priorytecie 2 i 1 są uporządkowane według polityki FIFO.

Kiedy w systemie pojawia się nowe żądanie, kolejność żądań oczekujących w odpowiedniej kolejce nie zmienia się, natomiast pozycja nowego żądania w kolejce wyznacza jest następująco.

Niech a^{s_a} oznacza żądanie a należące do sesji s_a, oczekujące w kolejce Q_k, $k \in \{1, 2\}$. Zdefiniowane są następujące zbiory i podzbiory żądań oczekujących w kolejce Q_k w chwili n, kiedy nadchodzi nowe żądanie należące do sesji s:

$$Q_k(n) = \{a^{s_a} \in Q_k\}, \tag{2}$$

$$Q_{k\_2}(n) = \{a^{s_a} \in Q_k : P^{s_a}(n) \in \{2,3,4\}\}, \tag{3}$$

$$Q_{k\_3}(n) = \{a^{s_a} \in Q_k : (P^{s_a}(n) = 4) \vee [(P^{s_a}(n) = 3) \wedge (v^{s_a}(n) \geq v^{s}(n))]\}, \tag{4}$$

$$Q_{k\_4}(n) = \{a^{s_a} \in Q_k : (P^{s_a}(n) = 4) \wedge [(v^{s_a}(n) > v^{s}(n)) \vee [(v^{s_a}(n) = v^{s}(n)) \wedge (RFM^{s_a}(n) \geq RFM^{s}(n))]]\}. \tag{5}$$

Żądania w kolejce Q_k są uporządkowane i wykonywane zgodnie z numerami pozycji żądań w kolejce. Początek kolejki jest wyznaczony przez numer 1, a jej koniec w chwili n przez numer $|Q_k(n)|$, który odpowiada liczności kolejki Q_k w chwili n.

Numer pozycji w kolejce Q_k dla nowego przyjętego żądania, które pojawiło się w chwili n, jest obliczany następująco:

$$z_k(n) = \begin{cases} |Q_k(n)| + 1 \text{ dla } P^s(n) = 1, \\ |Q_{k\_2}(n)| + 1 \text{ dla } P^s(n) = 2, \\ |Q_{k\_3}(n)| + 1 \text{ dla } P^s(n) = 3, \\ |Q_{k\_4}(n)| + 1 \text{ dla } P^s(n) = 4. \end{cases} \tag{6}$$

gdzie $k = 1$ dla żądania HTTP i $k = 2$ dla żądania dynamicznego.

Pseudokod algorytmu KARO-Rev przedstawiony jest na rys. 2.

Skuteczność algorytmu KARO-Rev została przebadana metodą zdarzeniowej symulacji komputerowej. Na potrzeby badań symulacyjnych opracowano model obciążenia oraz model biznesowego serwisu webowego [3], bazując na aktualnych danych literaturowych dotyczących charakterystyki obciążenia serwisów webowych. Na podstawie opracowanego modelu zaprojektowano oraz wykonano oprogramowanie symulatora serwisu WWW, w którym zaimplementowano algorytmy KARO-Rev i FIFO. Program symulacyjny został napisany w języku C++ z wykorzystaniem pakietu do modelowania złożonych systemów, CSIM19.

Wyniki badań symulacyjnych, przedyskutowane w [1, 3], pokazują, że algorytm KARO-Rev zapewnia wysoką jakość usług serwisu pod względem biznesowo-zorientowanych miar wydajności, nawet kiedy obciążenie serwisu trzykrotnie przekracza jego maksymalną pojemność. Algorytm zapewnia wysokie przychody i wysoką jakość usług dla kluczowych klientów, jak również zróżnicowane czasy odpowiedzi dla stron webowych należących do sesji o różnej randze, czyli krótsze czasy odpowiedzi dla bardziej wartościowych klientów. Wyniki te są osiągane kosztem nieco niższej jakości usług dla sesji zwykłych klientów.

```
if (nowe żądanie HTTP x_ip^s)
  if (żądanie dotyczy obiektu HTML)
    aktualizuj atrybuty sesji s cs(n), RFMs(n), es(n), ls(n), vs(n);
    aktualizuj priorytet sesji s Ps(n) zgodnie z (1);
    //kontrola przyjęć
    if [((I1 ≤ obciążenie systemu < I2) and (Ps(n) == 1))
       or ((obciążenie systemu ≥ I2) and ((Ps(n) == 1) or (Ps(n) == 2))]
       odrzuć żądanie;
    else
       przyjmij żądanie;
    end if
  else //żądanie dotyczy obiektu zagnieżdżonego
    przyjmij żądanie;
  end if
  //szeregowanie w kolejce Q1
  if (żądanie zostało przyjęte)
    wstaw żądanie do kolejki Q1 na pozycję wyznaczoną zgodnie z (6);
  end if
else if (nowe żądanie dynamiczne y_jp^s)
  // szeregowanie w kolejce Q2
  wstaw żądanie do kolejki Q2 na pozycję wyznaczoną zgodnie z (6);
end if
```

Rys. 2. Pseudokod algorytmu KARO-Rev, wykonywanego w chwili n
Fig. 2. Pseudocode of KARO-Rev algorithm being executed at the nth moment

5. Wnioski

W artykule przedstawiono nowe podejście do obsługi żądań użytkowników w serwisie WWW dla aplikacji e-commerce typu B2C, zorientowane na zapewnienie wysokiego przychodu dla właściciela e-sklepu przy jednoczesnym faworyzowaniu kluczowych klientów. Przedstawione podejście uzupełnia i rozszerza dotychczasowe badania w dziedzinie QoWS, przedyskutowane np. w [3], m.in. proponując zaimplementowanie metody identyfikacji i oceny kluczowych klientów, analizy RFM, w mechanizmie obsługi żądań dla biznesowego serwisu webowego.

Wyniki symulacyjne potwierdzają skuteczność proponowanego podejścia w porównaniu z tradycyjnym sposobem obsługi użytkowników w serwisie WWW. W ramach przyszłych prac planowane jest rozbudowanie podejścia o metody eksploracji danych, a także weryfikacja jego skuteczności przy użyciu prototypowego środowiska testowego.

6. Literatura

[1] Borzemski L., Suchacka G.: Discovering and Usage of Customer Knowledge in QoS Mechanism for B2C Web Server Systems. LNAI, Vol. 6277, Springer, Heidelberg, 2010, pp. 505-514.

[2] Suchacka G., Borzemski L.: Business-driven QoS Management of B2C Web Servers. LNCS, Vol. 6236, Springer, Heidelberg, 2010, pp. 93-100.

[3] Borzemski L., Suchacka G.: Business-oriented Admission Control and Request Scheduling for E-Commerce Web Sites. Cybernetics and Systems, Vol. 41, No. 8, 2010, pp. 1-17.

otrzymano / received: 25.11.2010
przyjęto do druku / accepted: 03.01.2011 *artykuł recenzowany*