Architektura warstwowa aplikacji internetowych

Józef E. Sienkiewicz[1,2], Paweł Syty[2]

[1]Instytut Informatyki Stosowanej, Państwowa Wyższa Szkoła Zawodowa w Elblągu
[2]Katedra Fizyki Teoretycznej i Informatyki Kwantowej, Wydział Fizyki Technicznej i Matematyki Stosowanej, Politechnika Gdańska

Wstęp

Historia Internetu i WWW zaczęła się na przełomie lat osiemdziesiątych i dziewięćdziesiątych, jako płaszczyzna wymiany informacji pomiędzy naukowcami zatrudnionymi w Europejskim Centrum Badań Jądrowych CERN. W prowadzonych tam badaniach z zakresu fizyki wysokich energii brało i bierze udział wiele osób z różnych krajów. W 1989 fizyk Tim Berners–Lee przedstawił pomysł sieci powiązanych ze sobą dokumentów opisujących m.in. projekty i wyniki prowadzonych badań. Pierwszy prototyp takiej sieci, bazujący na trybie tekstowym powstał dwa lata później. W 1999 roku MIT i CERN podpisały porozumienie w celu założenia organizacji pod nazwą World Wide Web Consortium, dające początek ogólnoświatowej sieci, do której początkowo dołączyły jedynie uniwersytety i firmy z branży informatycznej [1]. Obecnie Internet jest potężną siecią łączącą olbrzymią ilość nowoczesnych i dostarczających całego szeregu usług aplikacji WWW używanych przez indywidualnych klientów, małe i średnie przedsiębiorstwa jak również całe koncerny. Niektórzy uważają ogólnoświatową strukturę WWW jako pewnego rodzaju oprogramowanie znajdujące się w połowie drogi pomiędzy systemem informatycznym a systemem hipermedialnym [1, 2] . Najważniejsze cechy takiej hybrydy sprowadzają się do zarządzania ustalonymi strukturami danych w bazach danych jak i danymi typowymi dla multimediów, gdzie kompresja i efektywne przesyłanie odgrywają zasadniczą rolę. Prócz tego nawet się specjalnie nad tym nie zastanawiając, wymagamy od aplikacji internetowych użytecznej i na wysokim poziomie grafiki oraz aktywnego wspomagania naszych działań.

Wraz z nowym znaczeniem Internetu i aplikacji webowych rośnie też potrzeba rozwoju efektywnych metod wytwarzania tych aplikacji. Co prawda używa się tutaj tych samych technologii jak przy budowie innych produktów inżynierii oprogramowania, jednak pewne specyficzne wymagania, jak wyjątkowo silny nacisk na szybkie dostarczanie prototypów, a później oczekiwania niezawodnego działania w ciągu pełnych 24 godzin na dobę powodują, że pewne podejścia inżynierii oprogramowania jak budowanie przyrostowe i korzystanie ze wzorców architektonicznych są bardzo pomocne [3, 4]. W szczególności gotowe wzorce architektoniczne lub ich zręby znacznie przyspieszają przejście do następnych etapów budowy oprogramowania. Ułatwiają też szacowanie kosztów wytwarzania aplikacji metodami od ogółu do szczegółu (np. przy użyciu metody COCOMO) i od szczegółu do ogółu po rozpisaniu zadań dla członków zespołu. W dalszym toku prac, te oszacowania stają się podstawą realistycznego harmonogramu przedsięwzięcia.

W pracy przedstawiono wybrane abstrakcyjne podejście do wzorców architektonicznych opartych o modele warstwowe. W zależności od przeznaczenia aplikacji internetowej, a tym samym jej złożoności, są modele dwu-, trzy- i czterowarstwowe.

1. Podstawowe pojęcia i przykłady nowoczesnych aplikacji internetowych

WWW (*ang. World Wide Web*) - architektoniczna podstawa dostępu do powiązanych ze sobą dokumentów umiejscowionych na komputerze podłączonym do Internetu. Każdy taki dokument nosi nazwę strony WWW.

URL (*ang. Uniform Resource Locator*) – schemat jednoznacznego identyfikowania adresów komputerowych (IP), co w przypadku systemów rozproszonych posiada fundamentalne znaczenie. Odpowiedzi systemu na wezwania poszczególnych obiektów pozwalają na ich jednoznaczną identyfikację. URL jest szczególnym przypadkiem nadrzędnego standardu identyfikowania dokumentów w sieci – URI (*ang. Uniform Resource Identifier*).

HTML (*ang. HyperText Markup Language*) – służy do pisania dokumentów, zawiera instrukcje formatowania na podstawie których przeglądarka przetwarza zawartość dokumentów oraz wiąże je z innymi dokumentami. Rozwój HTML -a jest nierozłącznie związany z rozwojem przeglądarek. HTML najpierw zawierał jedynie instrukcje dotyczące sposobów prezentacji dokumentów. Obecnie pod nazwą XHTML oprócz wszystkich opcji HTML-a kryją się inne wyspecjalizowane znaczeniowo języki.

HTTP (*ang. TyperText Transmission Protocol*) jest protokołem komunikowania się pomiędzy stronami WWW, pozwala on komputerowi klienta zażądać przesłania danych i dokumentów znajdujących się na serwerze. Najczęściej używanymi operacjami są Get, która umożliwia przesyłanie danych z wybranego adresu URL, oraz Post, umożliwiająca przesyłanie danych do wybranego adresu URL.

Google Docs przeszukuje strony WWW w celu znalezienia żądanych słów kluczowych, wspomaga przetwarzanie tekstów, które przechowuje w odpowiednich formatach na serwerach Google'a. Teksty są zróżnicowane pod względem różnego stopnia udostępniania innym użytkownikom.

Wikipedia stała się jedną z najpopularniejszych stron WWW i obecnie stanowi źródło wielu pożytecznych informacji od ściśle naukowych do wszelkich bieżących. Zasadniczą cechą Wikipedii jest to, że jest redagowana przez swoich użytkowników.

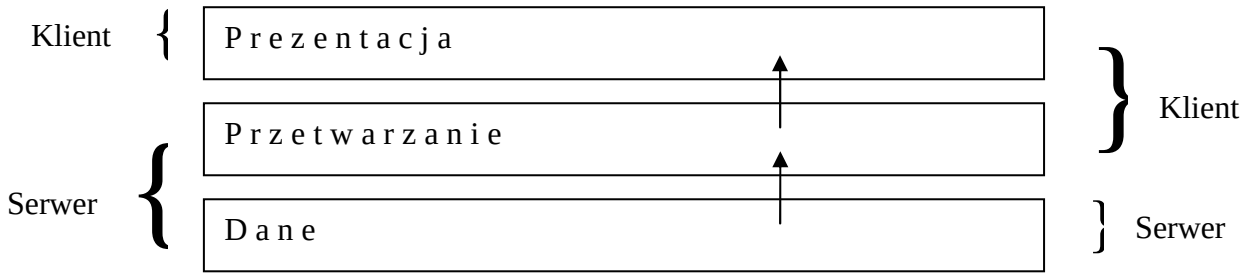
Flickr jest stroną WWW umożliwiającą wymianę fotografii. Zasada rozwoju jak w Wikipedii, im więcej użytkowników tym większy rozwój strony.

Nasza Klasa jako forum wymiany informacji pomiędzy zainteresowanymi użytkownikami. Swój sukces zawdzięcza pomysłowi. Przestrzeń utworzona przez jednego użytkownika jest otwarta dla innych.

Allegro – serwis aukcyjny, umożliwiający handel towarami i usługami w Internecie.

2. Model architektoniczny aplikacji WWW dla systemu klient-serwer

Architektura systemu klient–serwer opiera się na dwóch zbiorach obiektów rozproszonych czyli serwerach [5], które oferują różne usługi i klientach, którzy z tych usług korzystają. Klienci znają adresy potrzebnych im serwerów. Na rys. 1 pokazaliśmy ogólną warstwową architekturę systemu klient-serwer z dwoma różnymi podziałami warstw na procesy klienta i serwera. Najbardziej wewnętrzna warstwa odpowiada wszystkim operacjom serwera wykonanym na bazie danych. Warstwa środkowa jest odpowiedzialna za takie przetwarzanie danych, aby mogły być przeniesione do zewnętrznej warstwy w celu ich prezentacji na monitorze klienta. Warstwa zawierająca oprogramowanie przetwarzające może należeć do serwera (por. lewa strona rysunku) lub do klienta (por. prawa strona rysunku). Mamy wtedy odpowiednio do czynienia z modelem tzw. klienta cienkiego lub klienta grubego.

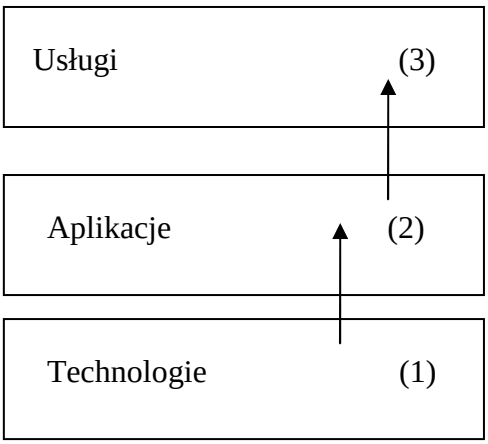


Rys. 1 Reprezentacja warstwowa modelu klient-serwer z różnymi podziałami warstw pomiędzy klientem a serwerem

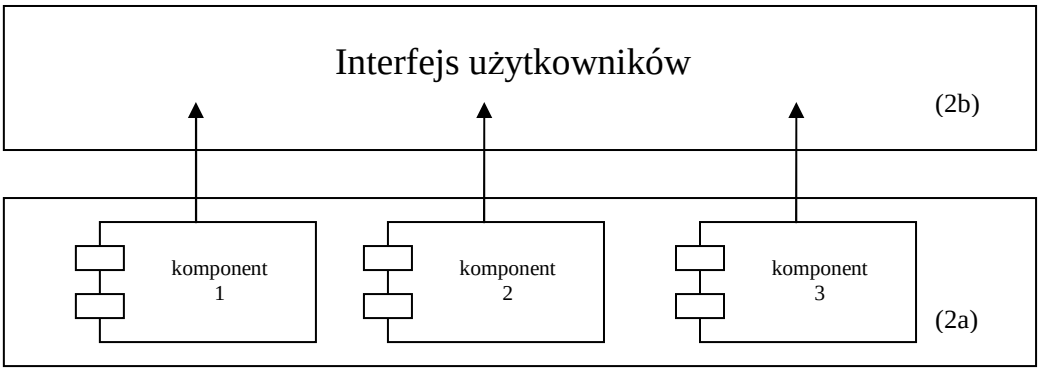
Pomimo ciągłego rozwoju, podstawowe zasady działania modelu klient-serwer pozostają praktycznie bez zmian. Łączy on rozproszone obiekty spełniające tradycyjne funkcje klientów lub serwerów, przy czym możliwa jest częsta zmiana tych funkcji bądź też jednoczesne pełnienie obu. Obecnie, coraz częściej aplikacje oparte o model cienkiego klienta ustępują powoli modelowi klienta grubego. Powoli odchodzi się od klienta wyposażonego jedynie w przeglądarkę, która otrzymuje dane gotowe do prezentacji, a które już poprzednio zostały przetworzone przez serwer. W tym przypadku oprogramowanie klienta służy jedynie do interpretacji HTML-a. Jednakże coraz większe obciążenie serwerów i związany z tym niedogodności powodują przechodzenie w kierunku modelu klienta grubego. Szczególną tutaj rolę odgrywa rozwój języka JavaScript, umożliwiającego dokonanie szeregu operacji po stronie klienta a wykorzystywanych dotychczas po stronie serwera i jednocześnie pozwalającego na rozwój interakcji. Ponadto, rozwój nowych technologii opartych na języku JavaScript jak np. AJAX (*ang. Asynchronous JavaScript and XML*) pozwala na przeprowadzanie szeregu operacji w tle na komputerze klienta, co zapewnia zwiększoną płynność interakcji z serwerem oraz dostosowanie do indywidualnych potrzeb klienta.

Samo dostarczania usług klientowi jest przedstawione na rys. 2 i 3 gdzie wymieniono jedynie przykładowe komponenty aplikacji internetowej. Do ich napisania często wykorzystuje się różne języki programowania i są one wykonane na różnych maszynach. Wewnętrzna warstwa technologiczna (por. rys. 2) oferuje usługi przetwarzanie-przechowywanie i dostarcza kanałów informacyjnych potrzebnych do uruchomienia aplikacji znajdujących się w wyższej warstwie. Warstwa technologiczna jest oparta o sprzęt

komputerowy i specjalistyczne oprogramowanie. Warstwa aplikacji w bezpośredni sposób wspomaga warstwy biznesowej i jest realizowana przez oprogramowanie użytkowe (aplikacyjne) [6, 7]. Najbardziej zewnętrzna warstwa usługowa bądź biznesowa oferuje potencjalnym klientom produkty i usługi mające wpływ na przebieg dotyczących ich procesów ekonomicznych. Warstwę aplikacyjną można poddać dalszej analizie i rozróżnić dwie podwarstwy (rys. 3), z których jedna jest zbudowana z określonych komponentów bądź podsystemów, co może sugerować przy budowie takiej aplikacji gotowych modułów [8]. Druga warstwa reprezentuje zbiór interfejsów do tych komponentów, przez które następuje możliwość dostarczania usług zainteresowanym klientom.



Rys. 2 Ogólny model warstwowy serwisu internetowego: (1) warstwa technologiczna, (2) warstwa aplikacyjna, (3) warstwa usługowa



Rys. 3 Podział warstwy aplikacyjnej z rys. 2 na warstwę komponentową (2a) i warstwę interfejsu użytkowników (2b)

3. Wymagania funkcjonalne i niefunkcjonalne. Ślady wymagań w planach testów

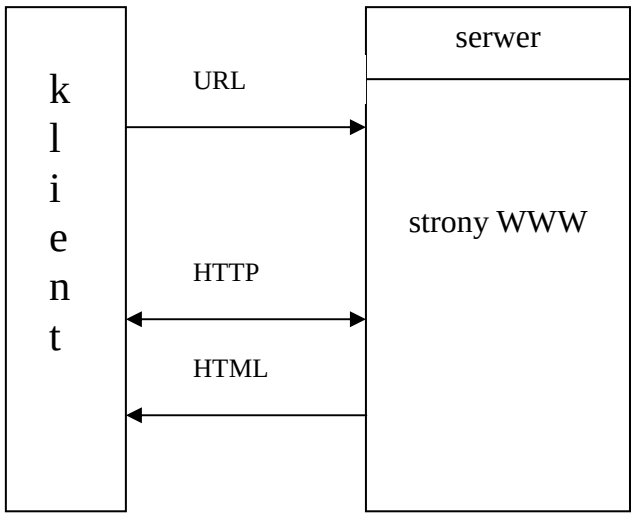
Wymagania funkcjonalne zależą w ścisły sposób od przeznaczenia aplikacji, które mogą być typowymi aplikacjami do zarządzania stronami WWW, aplikacjami typu bazodanowego, w tym hurtowniami wiedzy i wcześniej wspomnianymi aplikacjami

biznesowymi. W czasie sporządzania listy wymagań funkcjonalnych przygotowujemy odpowiednie plany testów zatwierdzających, gdzie należy uwzględnić specyfikacje aplikacji internetowych. Można tutaj wyróżnić różne modele testów np. testy modeli architektonicznych, testy poszczególnych komponentów i ich grup czyli podsystemów oraz typowe testy przebiegu procesów [9].

Główne wymagania niefunkcjonalne wobec aplikacji WWW dotyczą takich własności jak efektywność, skalowalność, kompatybilność, dostępność, użyteczność, bezpieczeństwo i ochrona przed ingerencją osób niepożądanych. Również tutaj weryfikacja każdego wymagania niefunkcjonalnego prowadzi do sporządzenia odpowiednich planów testów. Testy sprawdzające kompatybilność dotyczą uruchamiania aplikacji pod różnymi systemami oprogramowania oraz przy wykorzystaniu różnych przeglądarek internetowych. Testy dostępności sprawdzają, jakie typy użyteczności aplikacji są utrzymane przy redukcji konfiguracji sprzętowej.

4. Architektoniczne modele warstwowe aplikacji webowych

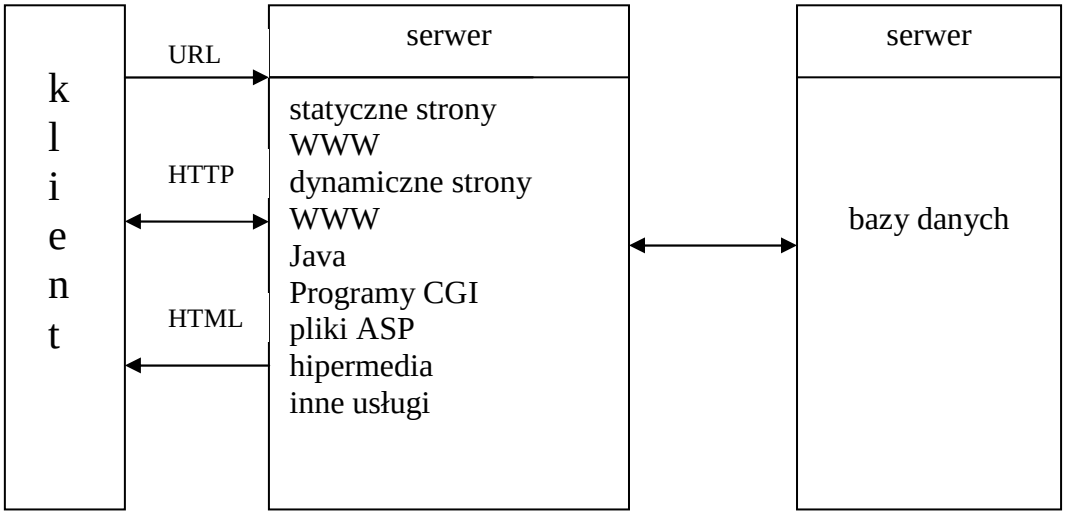
W zależności od przeznaczenia systemu należy dobrać odpowiedni model warstwowy przygotowywanej aplikacji [10]. Model ten powinien zależeć od stawianych wymagań funkcjonalnych. W najprostszym przypadku mamy do czynienia z aplikacją, której zadanie będzie polegało jedynie na przechowywaniu zawartości stron WWW z ewentualną ich aktualizacją. Architektura takiej aplikacji jest przedstawiona na rys. 4, gdzie te strony są tworzone i przechowywane w postaci plików HTML. W przypadku potrzeby aktualizacji stron dokonujemy odpowiedniej modyfikacji tych plików. Tego rodzaju architekturę można nazwać architekturą statyczna i dwuwarstwowa reprezentacja tej architektury jest jak najbardziej odpowiednia.



Rys. 4 Dwuwarstwowa architektura aplikacji dostarczającej strony WWW

W przypadku bardziej rozbudowanych wymagań takich jak dynamiczność stron WWW, zwiększona ich elastyczność i skalowalność, przechowywanie informacji w bazach danych bardziej stosowny jest model trzywarstwowy (rys. 5). Widać tutaj ważną rolę bazy danych wypełniającej jedną warstwę architektoniczną, która może być posadowiona na oddzielnym serwerze. Trzeba jednak zauważyć, że można tę warstwę i warstwę przetwarzania

umieścić na jednym serwerze. Idea tego typu architektury polega na tym, że dynamiczne strony WWW są za każdym razem tworzone na podstawie informacji przechowywanych w bazie danych.

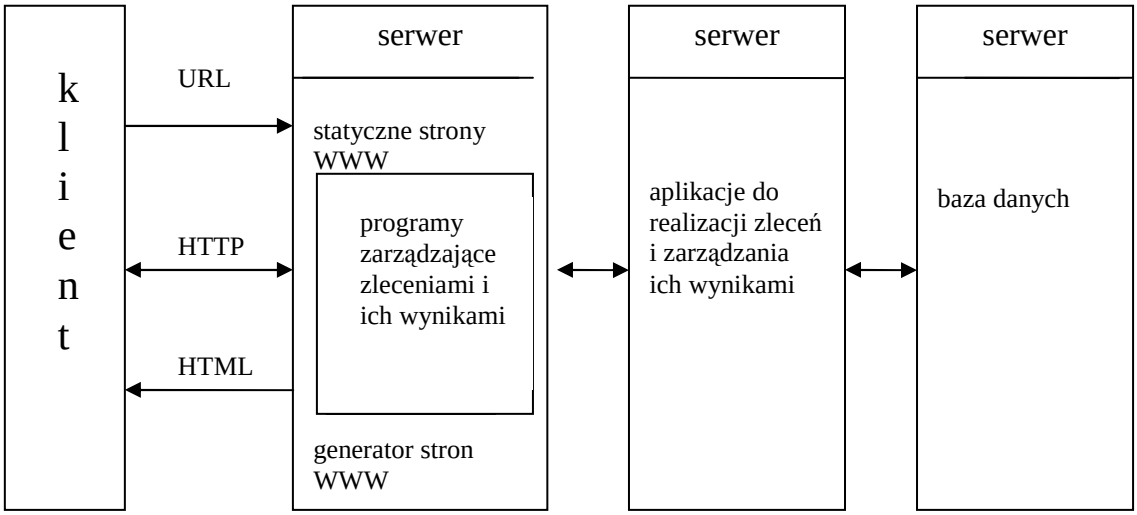


Rys. 5 Trzywarstwowa architektura aplikacji do przetwarzania danych

Warstwa przetwarzania odbiera zgłoszenia (czyli np. adres URL) od przeglądarki WWW umieszczonej po stronie klienta. W przypadku strony statycznej zapisanej w postaci języka HTML, po prostu odsyła ją do przeglądarki, a w przypadku złożonych serwisów najpierw następuje proces generowania strony. Najczęściej odbywa się to poprzez wypełnienie wcześniej przygotowanych szablonów HTML odpowiednią treścią, przy zastosowaniu wybranego skryptowego języka programowania (najpopularniejsze: PHP, Python, Ruby, ASP). Dodatkowo, w warstwie tej następuje komunikacja z serwerem bazy danych – przesłanie zapytania, odebranie danych wynikowych, ich obróbka i wygenerowanie na ich podstawie kodu HTML, przesyłanego następnie do przeglądarki. Transmisja pomiędzy klientem (przeglądarką) a warstwą przetwarzania odbywa się przy użyciu protokołu HTTP. Ostatnia warstwa, w której umieszczona jest baza danych, realizuje otrzymane zapytania (np. przekazane za pomocą języka SQL).

Główną zaletą architektury trzywarstwowej (inaczej trójwarstwowej) jest to, że pełną logikę aplikacji (czyli przetwarzanie danych) oraz komunikację ze stacjami klienckimi może realizować właśnie wydzielony serwer (czasem nazywany serwerem aplikacji), odciążając w ten sposób serwer bazy danych. Ponadto, serwer aplikacji może również wspomagać samo zarządzanie transakcjami bazodanowymi (tak się dzieje np. w platformie Tuxedo). Dla każdej aplikacji należy bardzo starannie rozplanować podział zadań pomiędzy serwery, dzięki czemu można zapewnić jej efektywną skalowalność i w przyszłości uniknąć problemów z wydajnością. O tym, jak jest to ważne, przekonali się w ostatnim czasie twórcy i użytkownicy serwisu Nasza Klasa, który wskutek nie zoptymalizowanego podziału zadań ciągle boryka się z problemem niedostatecznej wydajności, mimo ogromnego potencjału serwerów, na których serwis umieszczono. Programiści portalu nie przewidzieli tak ogromnej jego popularności i nie przerzucili odpowiedniej liczby zadań na serwery bazy danych, wskutek czego serwery aplikacji nie są stanie jednocześnie obsługiwać ogromnej liczby jednoczesnych połączeń ze stacji klienckich, komunikując się przy tym z bazą danych i otrzymując z niej gigabajty danych do przetworzenia i wysłania klientom. Z kolei serwis Allegro, równie popularny jak Nasza Klasa, jest przykładem dobrze zaprojektowanej architektury trzywarstwowej. Jego

znakomitą wydajność zapewniono poprzez starannie zaprojektowaną warstwę bazy danych, która realizuje wstępną obróbkę danych przekazywanych serwerowi aplikacji, odciążając go i tym samym umożliwiając mu bardzo efektywną komunikację z klientami. Należy w tym miejscu zauważyć, że zarówno Allegro jak i Nasza Klasa są posadowione na niemal identycznych platformach sprzętowych.



Rys. 6 Czterowarstwowa architektura aplikacji WWW do celów biznesowych

W przypadku bardziej rozbudowanych webowych aplikacji biznesowych, należy przeznaczyć dodatkową warstwę na oprogramowanie obsługujące procesy ekonomiczne. Warstwę tą, zgodnie z rys. 3 można dodatkowo podzielić na podwarstwy komponentową i interfejsową. W tym sensie nawiązuje ona do bardziej klasycznych architektur rozważanych w tradycyjnej inżynierii oprogramowania. W warstwie tej mogą się znajdować bardziej skomplikowane aplikacje biznesowe umożliwiające sprawne poruszanie się danej firmy w sferze ekonomicznej. W tym przypadku logiczne jest przedstawienie architektury jako modelu czterowarstwowego (rys.6). Wyraźne oddzielenie warstwy aplikacji biznesowych od pozostałych warstw pozwala na łatwiejszą ewolucję tego typu aplikacji. Np. dodając pewne oprogramowanie obsługujące procesy biznesowe pozostałe warstwy czyli prezentacji, przetwarzania danych i bazy danych pozostają nie zmienione.

5. Bezpieczeństwo warstwowych aplikacji webowych

Aby zapewnić bezpieczeństwo danych w internetowym systemie wielowarstwowym, należy zwrócić szczególną uwagę na autentykację i autoryzację użytkowników we wszystkich zastosowanych warstwach aplikacji, w szczególności w warstwie bazy danych. Autentykacja polega na identyfikacji oraz weryfikacji użytkownika, autoryzacja zaś na nadaniu użytkownikowi określonych uprawnień. Należy ponadto zapewnić transmisję danych między warstwami kanałami zapewniającymi jej poufność, np. przy użyciu protokołu SSL (*ang. Secure Socket Layer*) czy też IPSec. Już na etapie projektowania należy zwrócić szczególną uwagę na wrażliwe na atak rozwiązania i technologie, które mają być zastosowane w budowanej aplikacji, w celu minimalizacji ewentualnego zagrożenia. Przykładowo, należy starannie przemyśleć i zaplanować użycie kodu JavaScript na komputerze klienckim. Z jednej strony jest on bardzo prosty do przechwycenia (nie powinien zawierać więc jakichkolwiek

informacji związanych z bezpieczeństwem systemu – nazw użytkowników czy też haseł), z drugiej zaś strony może zostać użyty do wstrzyknięcia niepożądanego kodu do serwera aplikacji (np. za pośrednictwem formularza). Podobnie, źle zaprojektowany i zaimplementowany kod PHP może zostać użyty do uzyskania nieuprawnionego dostępu do informacji przechowywanych w bazie danych.

Podsumowanie

W pracy przedstawiliśmy trzy podstawowe wzorce architektoniczne dla aplikacji webowych. Użyty abstrakcyjny model warstwowy został przedstawiony dla trzech rodzajów aplikacji. Pierwsza aplikacja dostarczająca klientowi statyczne strony WWW posiada jedynie dwie warstwy. Bardziej skomplikowana trzywarstwowa architektura aplikacji dostarcza klientowi przetworzone dane przechowywane w warstwie bazodanowej. Wyróżniamy tutaj trzy warstwy: prezentacji, przetwarzania i bazy danych. Najbardziej skomplikowany wzorzec czterowarstwowy jest przeznaczony do projektowania aplikacji webowych mających na celu wspomaganie procesów ekonomicznych w firmach. Tutaj dodatkowa warstwa jest przeznaczona na oprogramowanie sterujące procesami ekonomicznymi.

Spis literatury

[1] M. Jazayeri, Some Trends in web Application Development, Future of Software Engineering (FOSE'07)

[2] A. E. Hassan and R. C. Holt, Architecture Recovery of Web Applications, Software Architecture Group (SWAG) Department of Computer Science University of Waterloo, Canada

[3] C. Standing, Methodologies for developing Web applications, Information and Software Technology 44 (2002) 151

[4] G. Neumann and U. Zdun, High-level design and architecture of an HTTP-based infrastructure for web applications, Word Wide Web 3 (2000) 13

[5] Sommerville, Inżynieria oprogramowania, WNT, Warszawa (2003)

[6] R. Kempkens, P. Rosch, L. Scott, J. Zettel, A multi-layer multi-view architecture for software engineering environments, Information and Software Technology 42 (2000) 141

[7] M. M. Lankhorst, Enterprise architecture modelling-the issue of integration, Advanced Engineering Informatics 18 (2004) 205

[8] C. Atkinson, Ch. Bunse and H.-Gerhard Grob, T. Kuhne, Towards a General Component Model for Web-Based Applications, Annals of Software Engineering 13 (2002) 3569

[9] G. A. Di Lucca, A. R. Fasolino, Testing Web-based applications: The state of the art and future trends, Information and Software Technology 48 (2006) 1172

[10] Li Jing-Feng, Li Yan, Chen Ping, A Unifield Architecture Model of Web Applications, Journal of Shanghai University (English Edition), 6 (2002) 221