STUDIA OECONOMICA POSNANIENSIA 2013, vol. 1, no. 2(251)

Agata Filipowska, Tomasz Kaczmarek, Krzysztof Węcel, Radosław Hofman

Uniwersytet Ekonomiczny w Poznaniu, Wydział Informatyki i Gospodarki Elektronicznej, Katedra Informatyki Ekonomicznej

Autor do korespondencji: Agata Filipowska, agata.filipowska@kie.ue.poznan.pl

ADAPTACYJNY SYSTEM BANKOWOŚCI KORPORACYJNEJ – ANALIZA WYMAGAŃ I WSKAZÓWKI IMPLEMENTACYJNE

Streszczenie: Adaptacyjne interfejsy użytkownika pozwalają modyfikować wygląd oraz sposób działania aplikacji, aby dostosować ją do preferencji użytkowników. Wykorzystuje je się często, gdyż pozwalają zwiększyć efektywność użytkowników w pracy z aplikacją oraz ułatwić im zaakceptowanie nowego systemu informatycznego. Adaptacyjność interfejsu napotyka jednak nowe wyzwania, jeżeli aplikacja jest tworzona w środowisku sieci WWW. Odpowiedzi na te wyzwania oraz ich rozwiązania zaprezentowano w artykule na przykładzie innowacyjnego, adaptacyjnego systemu bankowości elektronicznej. Architektura systemu została zaprojektowana w celu zapewnienia wydajności i skalowalności rozwiązania (możliwość obsługi nawet tysięcy użytkowników równocześnie). System monitoruje zachowanie użytkowników na podstawie zdarzeń generowanych przez nich w trakcie interakcji z systemem i wykorzystuje metody drążenia danych i analizy statystycznej w celu odkrywania preferencji użytkowników. Wzorce zachowań oraz preferencje są podstawą do utworzenia reguł adaptacji, które, uruchomione przez zmianę stanu aplikacji lub wygenerowane zdarzenia, powodują dostosowanie interfejsu aplikacji do potrzeb użytkownika.

Słowa kluczowe: personalizacja, adaptacyjność, architektura, system regułowy.

Klasyfikacja JEL: D14, L86, C8, C91.

Wstęp

Dostosowywanie aplikacji do potrzeb i zachowań użytkownika zawsze było jednym z czynników sukcesu i akceptacji dostarczanego oprogramowania [Brusilovsky 1998; Malinowski 1993]. Najczęściej wymaga się, aby użytkownik poznał i stosował reguły korzystania z interfejsu aplikacji, umożliwiając mu jedynie niewielkie jego dostosowanie (zmianę koloru, wielkości czcionki itp.). Użytkownicy coraz częściej jednak chcą mieć wpływ na wygląd i sposób działania systemu, z którego korzystają. W związku z tym wydaje się, że aplikacja zmieniająca się w reakcji na potrzeby

użytkownika, dopasowująca się do jego umiejętności, sposobu pracy i jednocześnie sugerująca akcje, które użytkownik powinien wykonać, znalazłaby szerokie uznanie na rynku oprogramowania, w szczególności oprogramowania wykorzystywanego do celów komercyjnych, na przykład w obszarze finansowym.

Artykuł prezentuje koncepcję systemu wykorzystującego techniki personalizacji do zmian interfejsu w codziennej pracy przez użytkowników korporacyjnych. Prowadzona personalizacja dotyczy nie tylko wyglądu aplikacji, ale także obszaru związanego z aktywnym wsparciem merytorycznym użytkownika, a więc sugestii akcji, które użytkownik zgodnie ze swoim dotychczasowym zwyczajem powinien w danym momencie wykonać.

Kolejne sekcje artykułu przedstawiają: analizę wymagań użytkownika i przykładowe funkcjonalności, jakie muszą być oferowane przez system, ogólny schemat systemu i przykładowe reguły personalizujące. Artykuł kończy się prezentacją przykładów dopasowania systemu do potrzeb użytkownika końcowego.

1. Personalizacja a adaptacyjność

Termin preferencja jest jednym z podstawowych pojęć w teorii ekonomii, odzwierciedlającym wybory, jakich dokonałby konsument, stojąc przed alternatywnymi możliwościami. W wypadku systemów informatycznych użytkownik wyraża preferencje odnośnie do wyglądu i funkcjonalności modelowanego systemu [Malinowski 1993]. Preferencje te mogą być wyrażane explicite (poprzez akceptację danej postaci systemu) lub implicite (gdy wynikają ze sposobu pracy z systemem).

Niezależnie jednak od sposobu wyrażenia, preferencje mogą być wykorzystywane w procesie adaptacji interfejsu rozumianej jako jego personalizacja (ang. *personalization*) lub dopasowanie (ang. *customization*).

Dopasowanie oznacza sytuację, w której użytkownicy mogą samodzielnie modyfikować wygląd strony. Wiele systemów umożliwia użytkownikom dostosowanie kolorystyki, wyglądu oraz prezentowanych funkcjonalności na interfejsie użytkownika GUI systemu. Podobnie jak dopasowanie, personalizacja jest procesem dostarczania relewantnej (odpowiedniej) informacji i funkcjonalności na podstawie indywidualnych preferencji użytkownika, wykorzystywanym jednak często w sytuacji, gdy użytkownik nie jest w stanie lub nie chce samodzielnie określić swoich potrzeb i system dokonuje tego na podstawie analizy zachowań.

Proces personalizacji obejmuje zwykle trzy fazy:

- poznanie i zrozumienie użytkownika – zgromadzenie informacji o użytkowniku i utworzenie opisującego go profilu; gromadzona informacja może dotyczyć jego zachowania we współpracy z systemem (kolejność otwierania okien, zatrzymywanie się, wpisywanie danych itp.),
- dostarczenie personalizowanych danych, stron, informacji itp.,

– określenie wpływu personalizacji na użytkownika i korygowanie strategii personalizacji pozwalające na reagowanie na opinie użytkownika dotyczące nowego, prezentowanego mu interfejsu (ang. *relevance feedback*) [Adomavicius i Tuzhilin 2005].

Opisane w literaturze metody personalizacji można podzielić na następujące grupy:

– metody heurystyczne wykorzystujące informacje o sposobie postępowania z systemem i na podstawie reguł zmieniających interfejs użytkownika,
– metody wykorzystujące model użytkownika zbudowany na podstawie informacji o jego zachowaniu, który następnie jest wykorzystywany w procesie personalizacji,
– metody wykorzystujące model użytkownika oraz informację o jego przynależności do określonej grupy użytkowników o podobnym sposobie pracy z systemem – pozwala to na dostosowywanie interfejsu, biorąc pod uwagę informacje o zaawansowaniu użytkownika, we współpracy z systemem (podobne błędy) – i podobnym sposobie postępowania itp.
– metody hybrydowe będące połączeniem powyższych [Adomavicius i Tuzhilin 2005; Atterer, Wnuk i Schmidt 2006; Mueller i Lockerd 2001].

W projekcie będą zastosowane właśnie metody hybrydowe pozwalające na najbardziej szczegółowe wykorzystanie informacji dotyczącej współpracy użytkownika z systemem.

2. Adaptacyjny System Bankowości Korporacyjnej

Adaptacyjny System Bankowości Korporacyjnej (ASBK) ma zapewnić użytkownikom wysoką produktywność w realizacji typowych operacji bankowych dzięki adaptacyjnemu interfejsowi użytkownika. System ma, na podstawie stale gromadzonych informacji o działaniach użytkownika, sugerować zmiany dostosowujące interfejs w taki sposób, aby wykonywanie zadań było dla użytkownika wygodniejsze i szybsze. Zwiększenie efektywności jest głównym celem działania systemu, a więc żadna adaptacja nie powinna mieć negatywnego wpływu na szybkość pracy użytkownika. System ma realizować te cele przez graficzne wyróżnianie najczęściej używanych elementów interfejsu lub dostosowanie kolejności elementów graficznych do sposobu ich użycia przez użytkownika, podpowiadanie wartości, które można wpisać w pola formularzy, automatyczne uzupełnianie danych, kiedy jest to możliwe, czy wreszcie przypominanie o powtarzalnych operacjach bankowych (np. wysyłanych cyklicznie przelewach).

Przyjęto, że system, realizując wspomniane wyżej sposoby personalizacji, będzie się posługiwał regułami określającymi, kiedy dane dostosowanie należy aktywować (podobne rozwiązanie można znaleźć w pracach R. De Virgilio, R. Torlone'a i G.J. Houbena [2006], P. Paskaleva [2009], P. Paskaleva i J. Serafimovej [2011] oraz

H. Wanga i in. [2011]). Reguły te mają być wykrywane w toku analizy zachowania użytkownika i dynamicznie zmieniane w odpowiedzi na zmiany w zachowaniu. Analiza zachowania jest prowadzona przy użyciu technik drążenia danych (np. wykrywania reguł asocjacyjnych [Agrawal i Srikant 1994]). Analogiczne badania opisano na przykład w pracy A. Kristofica i M. Belikovej [2005].

System ASBK jest nowoczesną aplikacją sieci Web (ang. *rich internet application*) utworzoną w technologii GWT i GWText (ver. 2.0.3)[1]. Dodatkowym ograniczeniem w projektowaniu systemu było to, że nie powinien on być oferowany w postaci dodatkowego oprogramowania instalowanego po stronie klienta ani wymagać korzystania z dodatkowego sprzętu.

Proponowany system obsługuje dwie kategorie użytkowników: użytkowników biznesowych aplikacji bankowej oraz administratora. Administrator systemu ASBK (zwany dalej również twórcą reguł) będzie odpowiedzialny za zarządzanie funkcjonowaniem rozwiązania.

W trakcie prowadzonych badań zidentyfikowano pięć podstawowych obszarów analizy funkcjonalnej systemu:

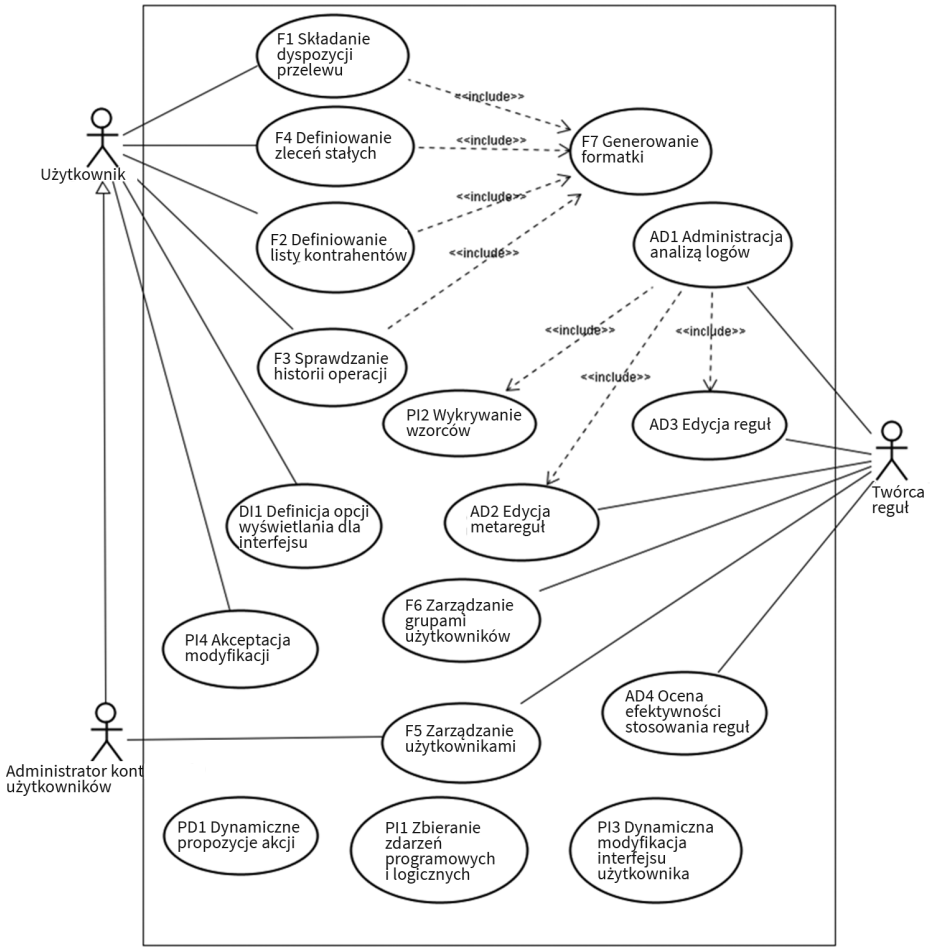
- ADMIN – wymagania dla funkcjonalności umożliwiających sprawną administrację systemem,
- FORMATKI – wymagania dla funkcjonalności formatek symulujących działanie aplikacji bankowej,
- CUSTOM/GUI – funkcjonalności dotyczące dopasowania interfejsu przez użytkownika systemu,
- PERSON/GUI – funkcjonalności personalizacji interfejsu,
- PERSON/DATA – funkcjonalności personalizacji danych.

Funkcjonalności, związki pomiędzy nimi oraz powiązania z różnego rodzaju użytkownikami systemu zaprezentowano za pomocą diagramu UML Use Case (rysunek 1). W kolejnych akapitach zaprezentowano krótki opis każdej z funkcjonalności.

Funkcjonalności AD1 i AD2 dotyczą zarządzania procesem analizy logów, w których zapisane są zachowania użytkownika. Celem tych funkcjonalności jest zapewnienie twórcy reguł wpływu na analizę zachowania użytkownika, której wynikiem jest utworzenie reguł personalizujących system. Reguły te są tworzone z wykorzystaniem metareguł, czyli szablonów dla reguł, które w toku analizy wypełniane są danymi wynikającymi z zachowania konkretnego użytkownika.

Funkcjonalność AD3 dotyczy edycji reguł. Wykryte reguły w toku analizy mogą zostać ręcznie zmienione przez twórcę reguł. Wynika to z potrzeby korekty możliwych niedoskonałości w regułach, które mogą się pojawić w wyniku analizy.

[1] GWT – Google Web Toolkit – ramy programowe pozwalające tworzyć aplikacje internetowe z wykorzystaniem asynchronicznych wywołań serwera WWW (technologia AJAX) w całości w języku Java. Szerszy opis na stronie http://code.google.com/intl/pl/webtoolkit/ [dostęp: 10.11.2012]. GWText – zestaw rozbudowanych kontrolek graficznych do wykorzystania w aplikacjach tworzonych w GWT.



Rysunek 1. Diagram Use Case dla ASBK

Celem analizy efektywności (funkcjonalność AD4) jest wybranie i wyłączenie reguł, które nie prowadzą do zwiększenia efektywności pracy użytkownika systemu. Analiza jest przeprowadzana okresowo i dotyczy procesów realizowanych przez użytkownika w systemie (F1–F4). Jeśli reguła wydłuża czas wykonania operacji, jest ona wyłączana ze stosowania.

Funkcjonalności F1, F2, F3, F4 są konieczne dla działania typowego systemu bankowego. Wszystkie używają funkcjonalności F7 (generowanie formatki). Funkcjonalność ta ma na celu wyświetlenie odpowiedniej formatki. Na jej wygląd mają wpływ informacje zawarte w profilu użytkownika, a więc między innymi przypisane mu reguły.

Funkcjonalności F5 (zarządzania użytkownikami) i F6 (zarządzanie grupami użytkowników) również są niezbędne dla działania typowego systemu bankowego.

Twórca reguł ma możliwość definiowania uprawnień dla grupy i przyporządkowania do niej użytkowników. Wykryte reguły dla użytkownika są ważniejsze niż reguły dla grup.

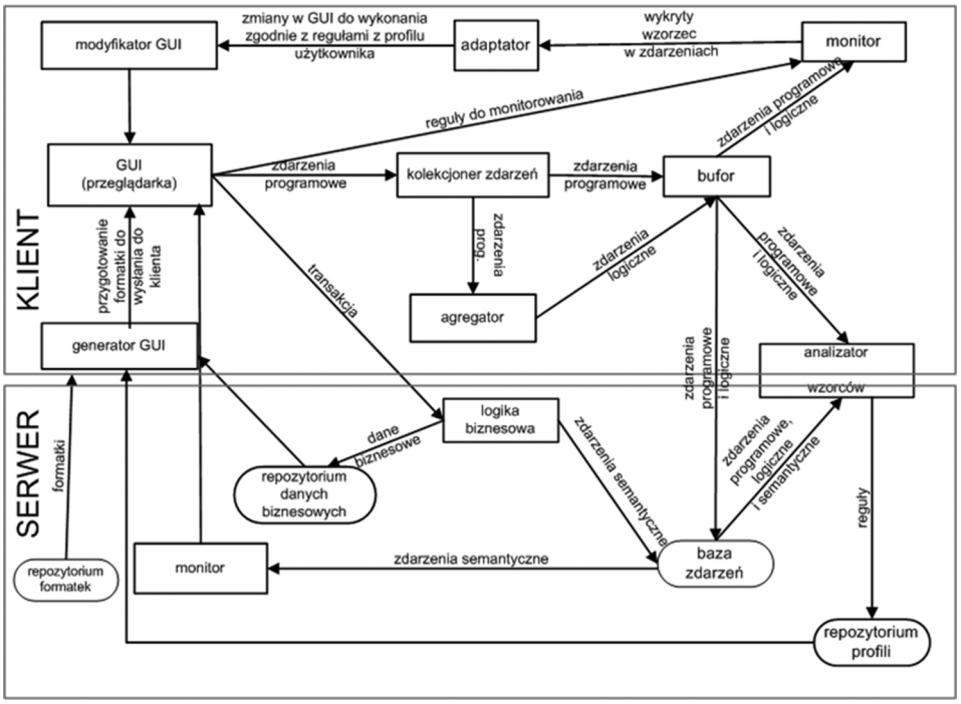
Użytkownik ma możliwość definiowania opcji dotyczących interfejsu (funkcjonalność DI1). Zapewnia to użytkownikowi wpływ na wygląd aplikacji. System ma zbierać informacje o zachowaniu użytkowników. W funkcjonalności PI1 system przechwytuje, rozpoznaje i zapisuje zdarzenia wygenerowane przez aplikację pod wpływem działań użytkownika. Funkcjonalność PI2 dotyczy wykrywania wzorców w zdarzeniach programowych, logicznych i semantycznych. Analiza logów przeprowadzana jest właśnie pod kątem wystąpienia wzorców zachowań użytkownika (grupy użytkowników). Na podstawie wzorców powstają reguły dotyczące zmiany zawartości i funkcjonalności interfejsu. Adaptacje w systemie są realizowane między innymi poprzez dynamiczne modyfikacje interfejsu użytkownika (funkcjonalność PI3). Zmiany w interfejsie są wprowadzane do interfejsu w reakcji na zajście zdarzenia, zbioru lub sekwencji zdarzeń określonych regułą. Funkcjonalność PI4 dotyczy akceptacji modyfikacji. Użytkownik ma możliwość wyłączenia reguł, które obniżają jego efektywność pracy, lub włączania reguł przez niego pożądanych, które wcześniej zostały wyłączone przez niego lub przez system.

Jednym ze sposobów na zwiększenie efektywności pracy użytkownika są propozycje akcji (PD1). Są one odpowiedzią na zajście części zdarzeń należących do wykrytego wcześniej wzorca zachowań. System sugeruje wykonanie akcji dążących do wypełnienia wzorca. Dla realizacji omówionych funkcjonalności zaproponowano architekturę systemu przedstawioną w kolejnej sekcji.

3. Architektura koncepcyjna systemu

Wysokopoziomowy widok architektury (rysunek 2) prezentuje przede wszystkim proces zbierania i analizy zdarzeń. Wybrany schemat architektoniczny to schemat klient–serwer. Uzasadnieniem dla tego wyboru jest wymaganie, aby system był aplikacją sieci Web.

Użytkownik po zalogowaniu, wykonując akcje w systemie, generuje zdarzenia programowe, logiczne i semantyczne. Najbardziej elementarne są zdarzenia programowe – związane z używaniem myszki i klawiatury na kontrolkach. Zdarzenia programowe są zbierane i przekazywane przez kolekcjoner zdarzeń do agregatora oraz bufora. Agregator rozpoznaje wśród ciągu zdarzeń programowych zdarzenia logiczne i wysyła je do bufora. Zdarzenia logiczne odnoszą się do zmian danych w polach formularzy: wprowadzenie, wyczyszczenie, potwierdzenie, zmiana, pominięcie pola. Dane o zdarzeniach przesyłane są na serwer i zapisywane w bazie zdarzeń. Używając reguł dotyczących użytkownika znajdujących się w profilu oraz zdarzeń zapisanych w buforach, monitor po stronie klienta wykrywa wystąpienie wcześniej uzyskanych wzorców, które następnie są przekazywane do adaptatora. Za dynamiczną modyfikację interfejsu użytkownika odpowiada modyfikator GUI.



Rysunek 2. Wysokopoziomowy widok architektury

System wychwytuje także zdarzenia semantyczne. Zdarzenie semantyczne jest to zdarzenie związane z operacjami na obiektach dziedzinowych, na przykład wyświetlenie historii rachunku, zatwierdzenie przelewu. Zdarzenia semantyczne są generowane po weryfikacji poprawności wykonania operacji przez logikę biznesową – wtedy zdarzenie jest przekazywane do bazy zdarzeń, a powiązane z nimi dane do repozytorium danych biznesowych. Monitor po stronie serwera rozpoznaje wzorce na podstawie zdarzeń semantycznych uzyskanych z bazy zdarzeń. Interfejs użytkownika GUI pobiera dynamiczne propozycje akcji z monitora po stronie serwera.

Analiza wzorców jest wykonywana na podstawie informacji zawartej w bazie zdarzeń oraz danych biznesowych. Na podstawie wykrytych wzorców zachowań tworzone są reguły. Generator GUI generuje odpowiednie formatki w odpowiedzi na żądanie przez użytkownika wyświetlenia okna.

4. Reguły w ASBK

Dostosowanie interfejsu następuje z wykorzystaniem reguł. Reguły w Adaptacyjnym Systemie Bankowości Korporacyjnej można podzielić na stanowe i zdarzeniowe. Reguła stanowa ma strukturę przedstawioną na rysunku 3.

<table>
<tr><td>IF</td><td colspan="3"><table><tr><td>attribute</td><td></td></tr><tr><td>value</td><td></td></tr></table> AND ... (pary atrybut-wartość powtórzone dowolną liczbę razy, odwołują się do atrybutów zawartych w pamięci roboczej)</td></tr>
<tr><td rowspan="6">THEN</td><td rowspan="6">multiple</td><td>action</td><td>set, show suggestion create, open (view), show alert create</td></tr>
<tr><td>ctrlid</td><td></td></tr>
<tr><td>businessObjType</td><td>(może być null lub transfer, directDebit itp.)</td></tr>
<tr><td>viewid</td><td></td></tr>
<tr><td rowspan="2"></td><td><table><tr><td>attribute</td><td></td></tr><tr><td>value</td><td></td></tr></table> AND ... (pary atrybut-wartość powtórzone dowolną liczbę razy, odwołują się do własności obiektu biznesowego lub kontrolki)</td></tr>
</table>

Rysunek 3. Struktura reguły stanowej

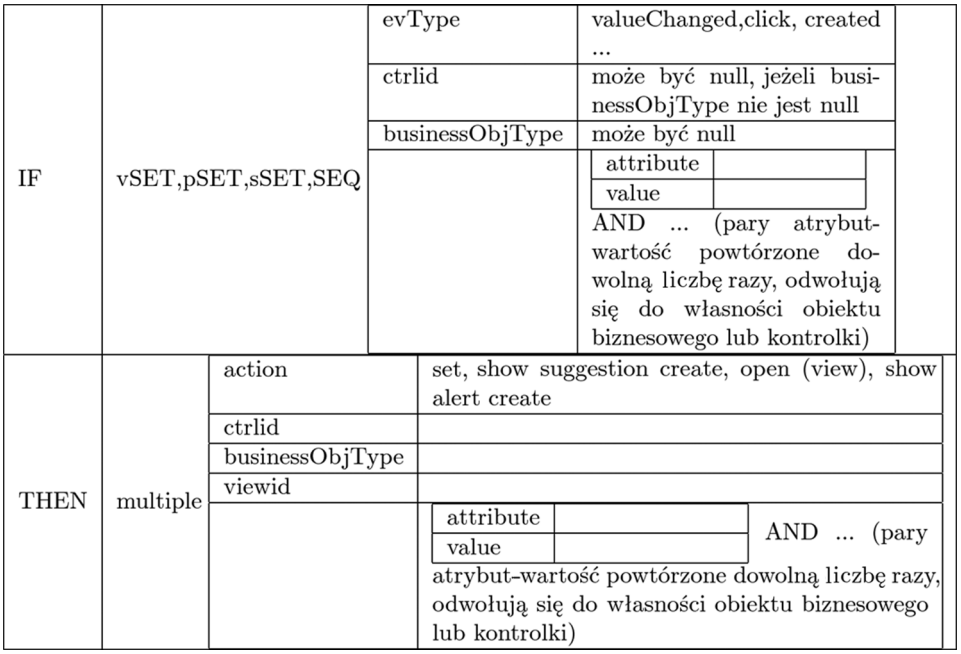
Pary atrybut–wartość odwołują się do pamięci roboczej. Termin ten jest zaczerpnięty z klasycznych systemów regułowych, w których reguły są odpalane na podstawie zawartości pamięci roboczej, a ich następniki modyfikują pamięć roboczą. W wypadku ASBK pamięć roboczą powinny tworzyć przynajmniej następujące elementy:

- currentTimeDD – bieżący dzień aktualizowany przy uruchomieniu aplikacji na podstawie zegara klienta,
- currentTimeMM – bieżący miesiąc, aktualizowany j.w.,
- currentTimeYYYY – bieżący rok, aktualizowany j.w.,
- currentviewid – aktualizowany za każdym razem, gdy tworzony jest widok szczegółowy, np. widok tworzenia nowego przelewu, potwierdzenie przelewu itd.,
- currentProcessId – unikatowy identyfikator aktualnego procesu; niweluje ryzyko uruchomienia reguły na przestrzeni dwóch różnych procesów biznesowych, kiedy jest to niepożądane.

Reguła zdarzeniowa ma strukturę przedstawioną na rysunku 4.

Główną cechą reguły zdarzeniowej jest to, że uruchamia się w reakcji na zachodzące zdarzenia: wystąpienie zarówno pewnego zbioru zdarzeń, jak i określonej sekwencji. Zdarzenia mogą wystąpić w ramach różnych tzw. koszyków lub w różnych okienkach czasowych. Dlatego też konieczne było wprowadzenia kwalifikatorów, które doprecyzowują zachowanie reguł:

- vSET (view SET) – reguła jest wrażliwa na zdarzenia występujące w ramach danego widoku, a dokładniej ostatniej (bieżącej) jego odsłony. Mechanizm wykonujący powinien dla takiej reguły działać następująco: powinien przeglądać zdarzenia dotyczące tylko bieżąco otwartego widoku i jeżeli są tam zdarzenia pasujące do elementów główki reguły, odpalać regułę. Po zmianie widoku na nowy wszystkie reguły, które nie uruchomiły się, powinny być wyczyszczone. Dodatkowo reguły, które już raz w danym widoku zostały odpalone (miały już raz dopasowane zdarzenia), powinny być oznaczone jako „wyczerpane", żeby nie uruchamiać reguł wielokrotnie.



Rysunek 4. Struktura reguły zdarzeniowej

- sSET (session SET) – jak wyżej, ale reguła jest wrażliwa na zdarzenia występujące w ramach całej sesji;
- vSEQ (view SEQUENCE) – reguła dotyczy sekwencji zdarzeń występujących w podanej kolejności, pomiędzy którymi mogą wystąpić inne nieokreślone zdarzenia. Kolejno napływające zdarzenia mogą pasować do kolejnych elementów sekwencji (następuje markowanie spełnionej częci główki reguły) lub nie (brak działań). Reguła jest rozpatrywana w ramach widoku; otwarcie nowego widoku resetuje regułę.
- sSEQ (session SEQUENCE) – jak wyżej, ale reguła jest rozpatrywana w ramach całej sesji; utworzenie nowej sesji resetuje regułę;
- SEQ (SEQUENCE) – reguła dotyczy ścisłej sekwencji zdarzeń, tj. każde kolejne zdarzenie albo pasuje, albo nie. Jeżeli pasuje, odpowiedni element reguły jest markowany, jeżeli nie, markowanie wszystkich elementów jest usuwane i dopiero wystąpienie pierwszego elementu główki może ponownie zacząć sekwencję.

Zastosowana typologia reguł łączy w sobie dwa zwykle stosowane podejścia – podejście klasyczne, oparte na bazie wiedzy (stanie), oraz podejście oparte na zdarzeniach. Takie połączenie jest wymagane, aby można było zrealizować różne scenariusze adaptacji systemu do potrzeb użytkownika (np. jeżeli dziś jest 15. dzień miesiąca, to przypomnij o przelewie zwykle wykonywanym tego dnia – reguła stanowa; jeżeli właśnie wypełniono pole odbiorcy przelewu, to podpowiedz

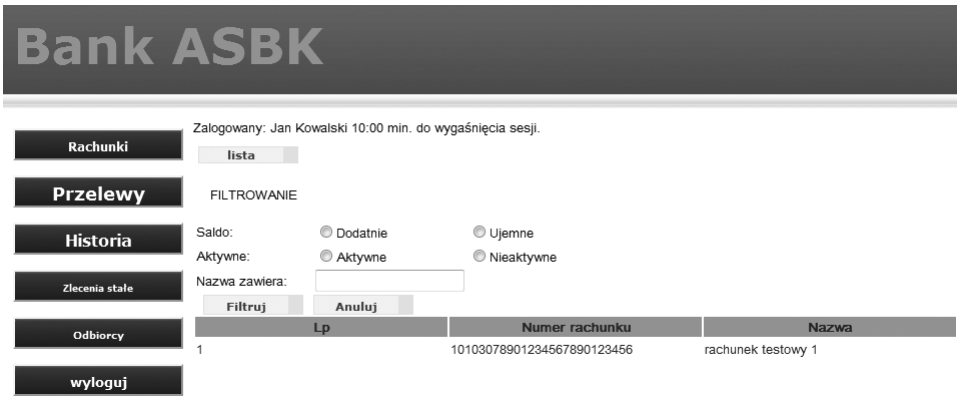
właściwy dla niego adres – reguła zdarzeniowa). Przyjęcie tylko jednego typu reguł za podstawowy nie pozwala zrealizować wszystkich scenariuszy – symulowanie obsługi zdarzeń za pomocą reguł stanowych jest skrajnie nieefektywne pamięciowo, natomiast symulowanie sytuacji związanych ze stanem aplikacji za pomocą reguł zorientowanych zdarzeniowo pociąga za sobą konieczność zapobieżenia problemom związanym z niepożądaną wielokrotną aktywacją reguł.

5. Przykład zastosowania

Działanie mechanizmów personalizujących w adaptacyjnym systemie bankowości korporacyjnej zostanie pokazane na przykładzie działania użytkownika wypełniającego polecenie przelewu. Zakładamy tutaj, że mamy do czynienia z użytkownikiem, który wcześniej wykorzystywał system do składania przelewów, i odkryto pewien zestaw reguł jego zachowania. Na przykład, system „nauczył" się, że najczęściej wybieraną przez użytkownika opcją są przelewy (rysunek 5). Przykładowa reguła zmieniająca kontrolkę wygląda następująco:

IF VIEW="Menu" THEN SETSTYLE(Przelew jednorazowy)="important"

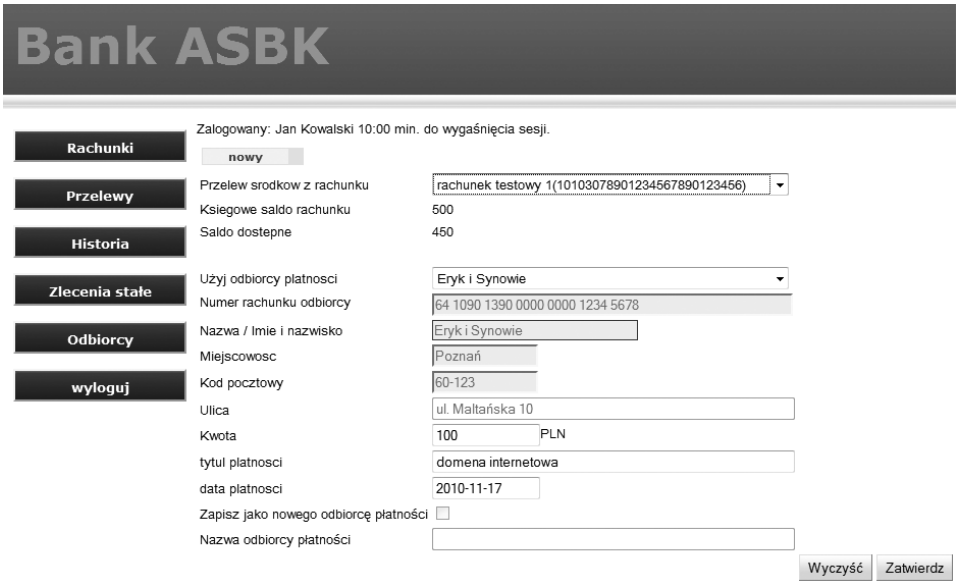
Reguła ta przy wyświetleniu przez aplikację menu zmienia wygląd przycisku „Przelew jednorazowy" i ustawia styl na wyróżniony („important").



Rysunek 5. Ekran użytkownika pozwalający na wybór opcji z menu z powiększonymi opcjami najczęściej wybieranymi przez użytkownika

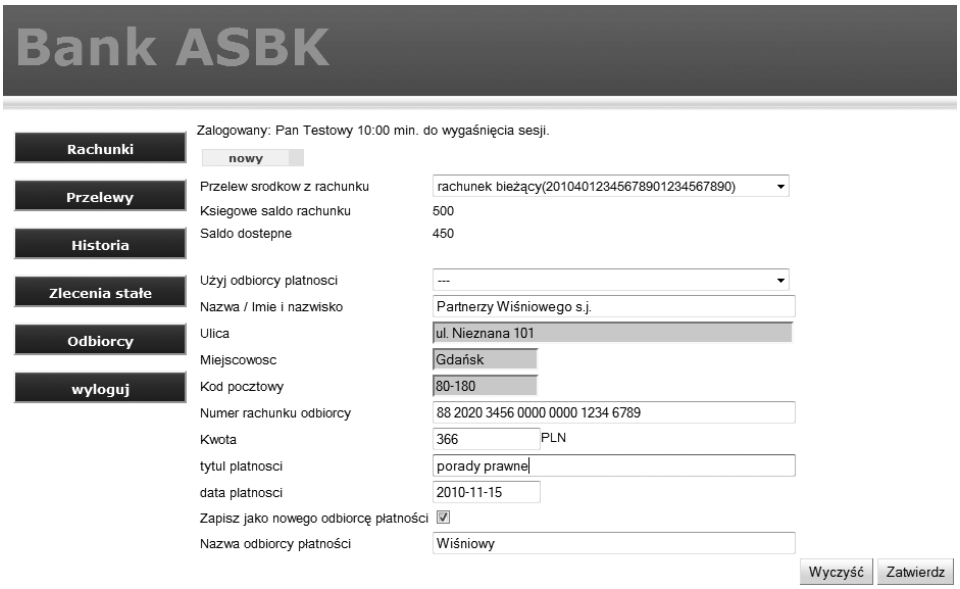
Źródło: Ekran programu

Po zalogowaniu użytkownik wybiera opcję dotyczącą składania przelewu. Zostaje mu wyświetlona standardowa formatka (widok) pozwalająca na złożenie kolejnego zlecenia przelewu, dostosowana jednak do jego potrzeb i wynikła z jego wcześniejszego zachowania (porównaj rysunki 6 i 7). Zmiany mogą tutaj dotyczyć



Rysunek 6. Standardowa formatka przelewu

Źródło: Ekran programu

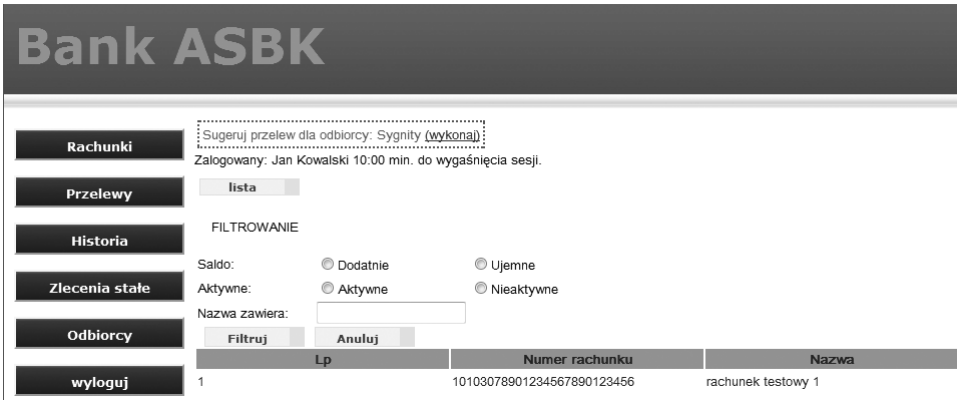


Rysunek 7. Formatka przelewu uwzględniająca sposób pracy użytkownika

Źródło: Ekran programu

kolejności kontrolek, wielkości lub ich koloru. Zmiany wprowadzane są z wykorzystaniem reguł, które na podstawie zachowania użytkownika zmieniają wygląd formatki.

W kolejnym kroku użytkownik wypełnia polecenie przelewu do kontrahenta Eryk i Synowie. Wypełniając polecenie przelewu, użytkownik generuje zdarzenia, które są zapisywane po stronie klienta, a następnie przekazywane na serwer. Będą one podstawą do analizy danych, której celem jest odnalezienie nowych wzorców zachowań użytkownika i utworzenie nowych reguł zarówno zmieniających wygląd kontrolek, jak i podpowiadających użytkownikowi zachowanie. Po wypełnieniu wszystkich pól, użytkownik weryfikuje dane i potwierdza przelew.



Rysunek 8. Interfejs zawierający główną formatkę wraz z odpowiednią informacją na pasku sugestii dotyczącą przelewu do kontrahenta

Źródło: Ekran programu

Użytkownik, realizując przelew dla poprzedniego kontrahenta (Partnerzy Wiśniowego Sp.j.), wywołał działanie wcześniej wykrytej reguły. Dotyczy ona wyświetlenia sugestii przelewu do kontrahenta Sygnity w odpowiedzi na wykonanie przelewu (rysunek 8). Jej postać jest następująca:

IF NEWTRANSFER „Partnerzy Wiśniowego Sp.j." THEN SUGGEST(NEWTRANSFER"SYGNITY S.A.")

Reguła powstała w procesie analizy logu zdarzeń semantycznych. Podczas analizy wykryto, że użytkownik po realizacji przelewu do kontrahenta Partnerzy Wiśniowego Sp.j. często wydaje polecenie przelewu do kontrahenta SYGNITY, stąd też sugestia wykonania podobnej akcji.

Podsumowanie

W artykule zaprezentowano koncepcję adaptacyjnego systemu bankowości korporacyjnej, mającego dawać nowe spojrzenie na kwestie elektronicznej bankowości. System, oprócz dostarczania klasycznych funkcjonalności, dostosowuje swój interfejs do działań użytkownika i tym samym zwiększa jego efektywność. Jeżeli użytkownik po zmianach będzie działał mniej efektywnie, system może automatycznie zdecydować o wycofaniu adaptacji.

Realizacja takiego systemu nie jest zadaniem trywialnym. Oprócz wyzwań wynikających z zastosowania wskazanej technologii tworzenia aplikacji, pojawiają się kwestie drążenia danych i odpowiedniego opracowania reguł tak, aby działanie dodatkowej „inteligentnej" warstwy systemu odbywało się w sposób niewidoczny dla użytkownika.

Bibliografia

Adomavicius, G., Tuzhilin, A., 2005, *Personalization Technologies: A Process-oriented Perspective*, Communications of the ACM, vol. 48, no. 10, s. 83–90.

Agrawal, R., Srikant, R., 1994, *Fast Algorithms for Mining Association Rules in Large Databases*, w: *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, Morgan Kaufmann Publishers Inc., San Francisco, s. 487–499, http://dl.acm.org/citation.cfm?id=645920.672836 [dostęp: 31.12.2012].

Atterer, R., Wnuk, M., Schmidt, A., 2006, *Knowing the User's Every Move: User Activity Tracking for Website Usability Evaluation and Implicit Interaction*, w: *Proceedings of the 15th International Conference on World Wide Web*, WWW '06, ACM, New York, NY s. 203–212, http://doi.acm.org/10.1145/1135777.1135811 [dostęp: 31.12.2012].

Brusilovsky, P., 1998, *Adaptive Hypertext and Hypermedia*, Springer.

De Virgilio, R., Torlone, R., Houben, G.J., 2006, *A Rule-based Approach to Content Delivery Adaptation in Web Information Systems*, w: *Proceedings of the 7th International Conference on Mobile Data Management*, MDM '06, IEEE Computer Society, Washington, DC, http://dx.doi.org/10.1109/MDM.2006.16 [dostęp: 31.12.2012].

Krištofič, A., Bieliková, M., 2005, *Improving Adaptation in Web-based Educational Hypermedia by Means of Knowledge Discovery*, w: *Proceedings of the Sixteenth ACM Conference on Hypertext and Hypermedia*, HYPERTEXT '05, ACM, New York, s. 184–192, http://doi.acm.org/10.1145/1083356.1083392 [dostęp: 31.12.2012].

Malinowski, U., 1993, *Adaptive User Interfaces: Principles and Practice*, Elsevier.

Mueller, F., Lockerd, A., 2001, *Cheese: Tracking Mouse Movement Activity on Websites, a Tool for User Modeling*, w: *CHI '01 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '01, ACM, New York, s. 279–280, http://doi.acm.org/10.1145/634067.634233 [dostęp: 31.12.2012].

Paskalev, P., 2009, *Rule-based GUI Modification and Adaptation*, w: *Proceedings of the International Conference on Computer Systems and Technologies and Workshop for*

PhD Students in Computing, CompSysTech '09, ACM, New York, s. 93:1–93:7, http://doi.acm.org/10.1145/1731740.1731841 [dostęp: 31.12.2012].

Paskalev, P., Serafimova, I., 2011, *Rule Based Framework for Intelligent GUI Adaptation*, w: *Proceedings of the 12th International Conference on Computer Systems and Technologies*, CompSysTech '11, ACM, New York, s. 101–108, http://doi.acm.org/10.1145/2023607.2023626 [dostęp: 31.12.2012].

Wang, H., Mehta, R., Supakkul, S., Chung, L., 2011, *Rule-based Context-aware Adaptation Using a Goal-oriented Ontology*, w: *Proceedings of the 2011 International Workshop on Situation Activity & Goal Awareness*, SAGAware '11, ACM, New York, s. 67–76, http://doi.acm.org/10.1145/2030045.2030061 [dostęp: 31.12.2012].

ADAPTIVE SYSTEM FOR CORPORATE BANKING: REQUIREMENTS ANALYSIS AND IMPLEMENTATION DETAILS

Abstract: Adaptive user interfaces allow users to modify the look and functionality of the application to suit their preferences. They are often used to increase user effectiveness and acceptance of a given software system. Adaptivity of the user interface poses certain new challenges if the application is delivered via the Web. The solutions for these challenges are presented in the article on the basis of an innovative adaptive e-banking system. The architecture of the system was designed for performance and scalability (up to thousands of users at the same time). The system is able to track user behaviour based on the events generated by the users during interaction with the system and uses data mining techniques to discover user preferences from the obtained event stream. The behaviour patterns and user preferences are converted into adaptation rules which are triggered either by change of the state of the application or events generated by the user.