BIULETYN INSTYTUTU AUTOMATYKI I ROBOTYKI
NR 29, 2010

System Wspomagania Pracy Nauczyciela

Piotr LITWINIUK, Jan CHUDZIKIEWICZ

Instytut Teleinformatyki i Automatyki WAT,
ul. Gen. S. Kaliskiego 2, 00-908 Warszawa
piotr.litwiniuk@gmail.com, jchudzikiewicz@wat.edu.pl

STRESZCZENIE: W artykule przedstawiono opis projektu oraz implementacji aplikacji wspomagającej pracę nauczyciela. Zdefiniowano wymagania funkcjonalne i niefunkcjonalne nakładane na system. Zaprezentowano diagramy interakcji wybranych przypadków użycia w notacji UML. Przedstawiono współpracę z zewnętrznymi, w stosunku do omawianego, systemami (SRU, SZZD) poprzez mechanizm Web Services. Omówiono wybrane rozwiązania implementacyjne wykorzystane w projektowanym systemie.

SŁOWA KLUCZOWE: aplikacja internetowa, usługi sieciowe, UML, ASP.NET, AJAX ASP.NET

1. Wprowadzenie

Proces nauczania, realizowany na uczelniach wyższych, wymaga od nauczycieli akademickich dokumentowania wyników pracy studentów. Rozwiązania tradycyjne, polegające na przechowywaniu ocen studenta z zaliczeń poszczególnych rygorów, wprowadzają duże niedogodności z ich przetwarzaniem. Chcąc ułatwić ten proces, zbudowano aplikację – ***System Wspomagania Pracy Nauczyciela*** (SWPN) [6], która stanowi elektroniczną wersję klasycznego dziennika nauczyciela. Korzyścią, wynikającą z używania SWPN, jest łatwy dostęp do informacji o wynikach uczniów, które są przechowywane w jednym centralnym miejscu.

Jednym z wymagań nakładanych na aplikację jest współdziałanie z ***Systemem Rejestracji Użytkowników*** (SRU) [6] oraz z ***Systemem Zdalnego Zarządzania Danymi*** (SZZD) [7] zaimplementowanym w postaci usługi sieciowej[1]

[1] **Usługa sieciowa** – część oprogramowania z niezależnym od platformy interfejsem, która może być automatycznie lokalizowana oraz wywoływana [1].

(ang. *Web Services*). SZZD dostarcza zestawu metod, za pomocą których SWPN komunikuje się z bazą danych przechowującą wyniki studentów. Zaletą wykorzystania usług sieciowych jest podział projektu na moduły, co znacząco ułatwiło realizację projektu oraz umożliwia dalszy rozwój aplikacji.

2. Wymagania funkcjonalne

Realizacja projektu wymagała opracowania zestawu wymagań, które system SWPN powinien spełniać. Wymagania te to:

1. Współdziałanie z SZZD oraz SRU.
2. Import/Eksport danych z/do pliku.
3. Możliwość sortowania wybranych danych.
4. Zarządzanie przedmiotami.
5. Zarządzanie użytkownikami.
6. Możliwość zarządzania wynikami.
7. Wybór sposobu oceniania.
8. Dodawanie ocen cząstkowych dla słuchaczy.
9. Wyznaczanie oceny końcowej.
10. Możliwość wyboru rodzaju zajęć.
11. Możliwość opisu każdej oceny bądź punktu.
12. Możliwość korzystania z aplikacji przez kilku wykładowców.

Jednym z wymagań, nakładanych na projektowany system, jest współdziałanie z SZZD w zakresie wymiany danych oraz z SRU w zakresie uwierzytelniania i autoryzacji użytkowników.

Kolejnym wymaganiem, jakie system powinien spełnić, jest możliwość odczytu danych z pliku. System umożliwia importowanie danych z pliku tekstowego. Wprowadzono tę właściwość do systemu, aby ułatwić użytkownikowi tworzenie list studentów. Pliki tekstowe są stosowane głównie jako pliki: *konfiguracyjne*, *dokumentacje* lub *instrukcje*. W opracowanym rozwiązaniu przyjęto następujący format danych tekstowych: *nazwisko:imię:numer_indeksu_studenta*, np. „Litwiniuk:Piotr:37464”. Dla identyfikacji poszczególnych grup znaków, jako znak rozdzielający, w opracowanym rozwiązaniu przyjęto znak – „**:**”.

Kolejnym rodzajem pliku, z którego aplikacja może importować dane, jest plik „xls”. Pliki tego typu przechowują dane arkusza kalkulacyjnego Microsoft Excel. SWPN daje możliwość pobrania z pliku xls następujących informacji: *nazw grup*, *nazw przedmiotów*, *danych personalnych studentów* oraz *nauczycieli*. Niemniej jednak uwzględniono także możliwość ręcznego wprowadzania danych.

SWPN implementuje funkcję eksportu danych do pliku. Pojęcie „eksport" jest rozumiane w tym kontekście jako umieszczenie tekstu, pobranego z aplikacji „dziennik nauczyciela", w pliku, z którego inny program mógłby odczytać informacje. Dane są zapisywane zarówno w plikach z rozszerzeniem „txt" oraz „xls", jak również w plikach zgodnych ze standardem „PDF". Jest to skrót od „*Portable Document Format*", oznacza format pliku opracowany i promowany przez firmę Adobe Systems, regulowany standardem ISO 15930. SWPN realizuje funkcję eksportu wyników studentów z wybranej grupy.

Kolejną funkcją realizowaną przez SWPN jest sortowanie. Sortowanie polega na uporządkowaniu zbioru danych względem pewnych cech charakterystycznych dla każdego elementu tego zbioru. SWPN klasyfikuje elementy alfabetycznie na podstawie: *nazwisk studentów*, *nazw grup*, *nazw przedmiotów*, *numerów indeksów*. Aplikacja ma za zadanie sortować wyświetlane dane w tabeli, rosnąco lub malejąco w zależności od wybranego przez użytkownika kryterium. Implementacja tej funkcji ułatwia proces sortowania, a także poprawia estetykę wyświetlania danych przez program. Informacje, zapisywane do pliku, są posortowane alfabetycznie, co stanowi duże udogodnienie przy ich przeszukiwaniu.

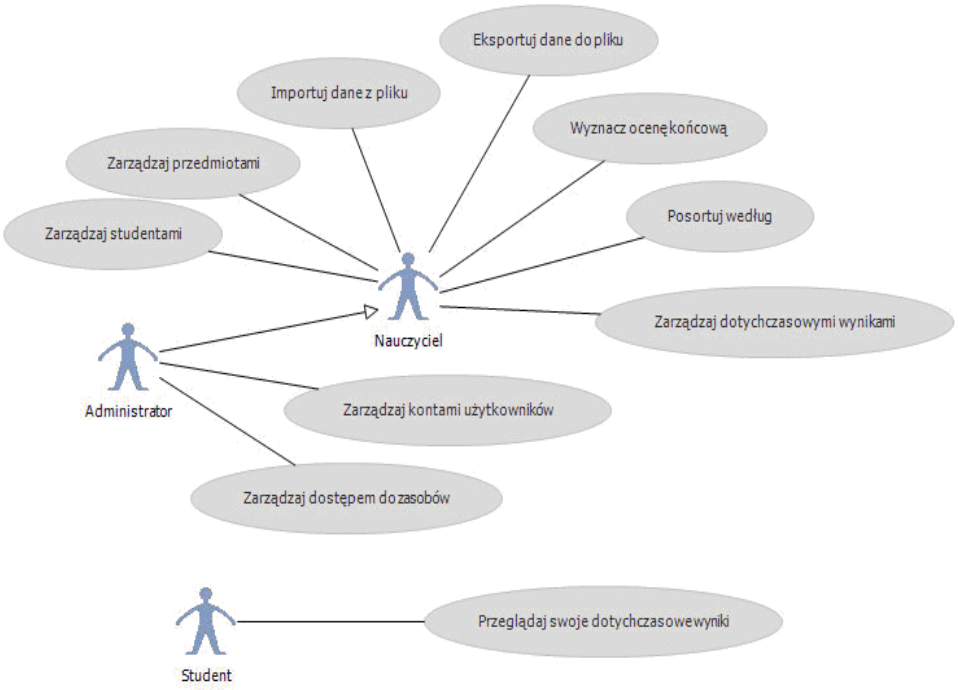
System SWPN umożliwia zarządzanie przedmiotami, polegające na: powiązaniu przedmiotu z wykładowcą, wprowadzeniu nazw przedmiotów do SZZD, usunięciu przedmiotów z SZZD i zmianie nazwy przedmiotu (edycji).

Kolejną funkcją realizowaną przez SWPN jest zarządzanie ocenami lub punktami uzyskanymi przez studenta. To wymaganie określa możliwość: wyboru sposobu oceniania, dodawania ocen (lub punktów) dla studentów, usuwania oraz edycji wyników. Każdą ocenę (lub punkt) nauczyciel może opisać w postaci „Laboratorium nr 1", „Sprawozdanie nr 3", „Sprawdzian z programowania", itp. Osoba nieuprawniona nie ma dostępu do edycji tych danych.

Na podstawie uzyskanych przez studenta wyników, SWPN może wystawić ocenę końcową z wybranego przedmiotu. Implementacja tego wymagania polega na obliczeniu przez program średniej arytmetycznej lub ważonej z osiągniętych przez studenta ocen. Ocena końcowa, wystawiona przez system, jest obliczana zgodnie ze zdefiniowaną przez wykładowcę skalą ocen. Proces ten jest wykonany automatycznie po zaznaczeniu przez wykładowcę opcji.

Wykładowca ma możliwość wybrania rodzaju prowadzonych zajęć poprzez zestaw opcji menu dostępnych w aplikacji.

Jednym z założeń, określonych przez specyfikację wymagań, było umożliwienie korzystania z systemu przez wykładowców z różnych uczelni. W takim przypadku należało uwzględnić możliwość występowania kilku takich samych numerów indeksów studentów. Mając powyższe na uwadze, w SWPN do identyfikacji studentów wykorzystywane są unikalne numery, niezwiązane z ich numerami indeksów. Numer nadany studentowi jest chroniony i nie mają do niego dostępu użytkownicy nieuprawnieni.

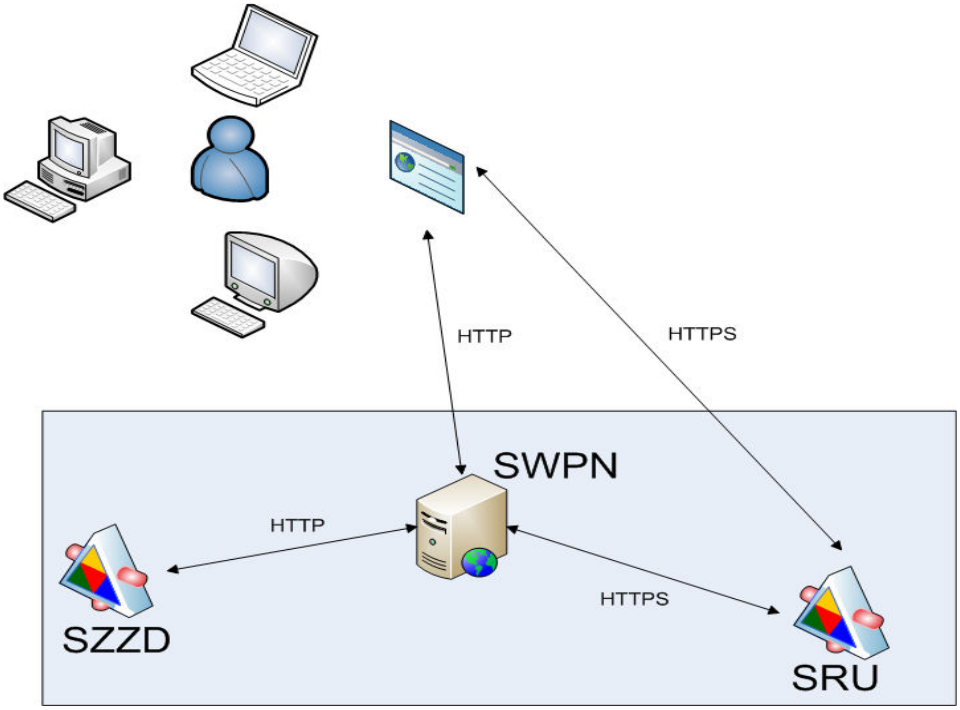


Rys. 1. Diagram przypadków użycia

Na rys. 1 przedstawiono wymagania funkcjonalne w postaci diagramu przypadków użycia w notacji UML. Wydzielono przypadki użycia: *zarządzaj użytkownikami*, *zarządzaj przedmiotami*, *zarządzaj dotychczasowymi wynikami*, *posortuj według*, *importuj dane z pliku*, *eksportuj dane do pliku*, *wyznacz ocenę końcową*, *zarządzaj dostępem do zasobów*, *przeglądaj dotychczasowe wyniki*. *Zarządzanie* w tym przypadku oznacza dodawanie, usuwanie oraz edycję danych. Aktorzy, którzy będą korzystać z systemu, to: *nauczyciel*, *administrator*, *student*. Administrator „dziedziczy" po nauczycielu, dzięki czemu uzyskuje dostęp do wszystkich funkcji nauczyciela.

3. Projekt systemu

Proces komunikacji systemu SWPN z bazą danych realizowany jest za pośrednictwem *Web Service*, który udostępnia zestaw metod pozwalających na manipulowanie danymi (odczyt/zapis z bazy danych). Aplikacja „dziennik nauczyciela" ma dwie referencje do usług sieciowych (rys. 2). Jedna jest wskazaniem na SZZD, druga natomiast na SRU.



Rys. 2. Model współdziałania aplikacji

Na rys. 2 przedstawiono model współdziałania pomiędzy trzema aplikacjami. Użytkownik, wpisując w przeglądarce adres, łączy się z aplikacją „dziennik nauczyciela". SWPN umożliwia użytkownikowi zalogowanie się do systemu po sprawdzeniu, czy dana osoba zalogowała się w SRU. W przypadku gdy użytkownik nie będzie zalogowany do SRU, SWPN automatycznie przekieruje odwołanie na stronę logowania do SRU. Zadaniem użytkownika jest zarejestrowanie się w *Serwisie Rejestracji Użytkowników* oraz zgłoszenie chęci uzyskania dostępu do aplikacji – *dziennik nauczyciela*. Autoryzację użytkownika w SWPN realizuje administrator SWPN lub upoważniona osoba. Ponieważ system wykorzystuje mechanizm ról do określania uprawnień dostępu, osoba przeprowadzająca autoryzację musi wykonać dowiązanie użytkownika do odpowiedniej roli – *student* bądź *nauczyciel*.

3.1. Opis SZZD i SRU

W dalszej części tego rozdziału przedstawiono opis wykorzystanej usługi sieciowej, która w implementacji została nazwana *Database_Service*. Dostęp do usługi zapewnia klasa pośrednika o nazwie *Service*, która udostępnia zestaw metod do zarządzania bazą danych. Przyjęto, że nazwy metod są tworzone po angielsku. W nawiasach zostały podane typy argumentów metod. Wszystkie metody, których zadaniem jest zwracanie danych, wykorzystują obiekt *DataSet*. Wykorzystywane przez SWPN metody klasy *Service*, to:

1. AddActivity(long, string, string, float, DateTime) – dodaj aktywność.
2. AddGroup(string) – dodaj grupę.
3. AddGroupLessons(int, int, int, int) – dodaj zajęcia dla grupy.
4. AddLessonsGradingScale(long, float, float, float, float, float) – dodaj skalę ocen dla studenta.
5. AddMark(long, float, string) – dodaj ocenę.
6. AddMessageByMailAddress(string, string, long, string[]) – dodaj wiadomość użytkownikowi o adresie.
7. AddMessageByUID(string, string, long, long) – dodaj wiadomość użytkownikowi wskazanemu za pomocą UID (ang. *user identifier*).
8. AddSchedule(long, DateTime, object[][]) – dodaj plan zajęć.
9. AddStudent(int, string, string, int) – dodaj studenta do grupy.
10. AddStudentsLesson(int, int, int, int) – dodaj zajęcia dla studenta.
11. AddSubject(string) – dodaj przedmiot.
12. AddTeacher(string, string, string) – dodaj nauczyciela.
13. AddTypeOfLesson(string) – dodaj rodzaj zajęć.
14. DeleteActivity(long) – usuń aktywność.
15. DeleteGroup(int) – usuń grupę.
16. DeleteLessonsGradingScale(long) – usuń skalę ocen.
17. DeleteMessage(long) – usuń wiadomość.
18. DeleteSchedule(long, DateTime) – usuń plan zajęć.
19. DeleteStudent(int) – usuń studenta.
20. DeleteStudentsLesson(long) – usuń zajęcia.
21. DeleteSubject(int) – usuń przedmiot.
22. DeleteTeacher(int) – usuń nauczyciela.
23. DeleteTypeOfLesson(int) – usuń rodzaj zajęć.
24. GetGroupList2() – zwróć listę wszystkich grup.
25. GetLessonsActivitiesList2() – zwróć aktywności wszystkich zajęć.
26. GetLessonsGradingScale(long) – zwróć skalę ocen.
27. GetLessonsList2() – zwróć wszystkie zajęcia.
28. GetSchedule(long, DateTime) – zwróć plan zajęć.
29. GetStudentsLessonsList2(int) – zwróć zajęcia dla studenta.
30. GetStudentsList2() – zwróć listę wszystkich studentów.
31. GetStudentsListInGroup(int) – zwróć listę wszystkich studentów w grupie.
32. GetSubjectsList2() – zwróć listę wszystkich przedmiotów.
33. GetTeachersLessonsList2() – zwróć listę wszystkich zajęć nauczycieli.

34. GetTeachersList2() – zwróć listę wszystkich nauczycieli.
35. GetTypeOfLessonsList2() – zwróć listę wszystkich rodzajów zajęć.
36. GetUsersReceivedMessages2(long) – zwróć wszystkie otrzymane wiadomości.
37. GetUsersSentMessages2(long) – zwróć wszystkie wysłane wiadomości.
38. Service() – konstruktor obiektu klasy *Service*.
39. SetLessonsFinalGrade(long) – wystaw ocenę końcową.
40. UpdateActivity(long, string, string, float, float, string, DateTime) – aktualizuj aktywność.
41. UpdateGroup(int, string) – aktualizuj nazwę grupy.
42. UpdateLessonsGradingScale(long, float, float, float, float, float) – aktualizuj skalę ocen.
43. UpdateSchedule(long, DateTime, object[][]) – aktualizuj plan zajęć.
44. UpdateStudent(int, string, string, int, int) – aktualizuj dane personalne studenta.
45. UpdateStudentsLesson(long, int, int, int, int) – aktualizuj zajęcia studenta.
46. UpdateSubject(int, string) – aktualizuj nazwę przedmiotu.
47. UpdateTeacher(int, string, string, string) – aktualizuj dane personalne nauczyciela.
48. UpdateTypeOfLesson(int, string) – aktualizuj nazwę rodzaju zajęć.

W SWPN wykorzystano także usługę, która w implementacji została nazwana *Authorization_Service*, nazwa kodowa to SRU. Dostęp do usługi zapewnia klasa pośrednika o nazwie *Auth*, która udostępnia zestaw metod do realizacji procesu uwierzytelnienia. Przyjęto, że nazwy metod tworzone są po angielsku. Komunikacja SWPN z tą usługą sieciową jest wykonywana poprzez protokół SSL[2], który szyfruje połączenia. *Authorization_Service* udostępnia mechanizm grup, który w SWPN odpowiada rolom przypisanym użytkownikom. Mechanizm grup charakteryzuje się drzewiastą strukturą. Poniżej przedstawiono metody SRU:

1. AddCurrentUserToGroup(string) – dodaj zalogowanego użytkownika do grupy.
2. AddUserToGroup(string, string) – dodaj użytkownika do grupy.
3. Auth() – konstruktor obiektu klasy *Auth*.
4. Authenticate(string, string, string) – uwierzytelnij użytkownika.
5. AuthenticateWithNoTicket(string, string) – uwierzytelnij użytkownika bez biletu.
6. AuthorisedUsers() – zwróć autoryzowanych użytkowników.
7. AuthoriseUser(string) – autoryzuj użytkownika.
8. CheckTicket(string) – sprawdź bilet.
9. CreateChildGroup(string, string) – utwórz grupę potomną.
10. CreateGroup(string) – utwórz grupę.
11. DeleteGroup(string) – usuń grupę.
12. GetAllGroups() – zwróć grupy.
13. GetAllUsers() – zwróć użytkowników.
14. GetChildrenForGroup(string) – zwróć grupy potomne dla wybranej grupy.

[2] **SSL** (ang. *Secure Socket Layer*) – protokół służący do bezpiecznej transmisji zaszyfrowanego strumienia danych opracowany przez firmę Netscape w 1994.

15. GetGroupsForCurrentUser() – zwróć grupy dla zalogowanego użytkownika.
16. GetGroupsForUser(string) – zwróć grupy dla wybranego użytkownika.
17. GetGroupsWithoutParentForCurrentUser() – zwróć grupy bez grup rodzicielskich dla zalogowanego użytkownika (są to grupy na szczycie hierarchii).
18. GetGroupsWithoutParentForUser(string) – zwróć grupy bez grup rodzicielskich dla wybranego użytkownika (są to grupy na szczycie hierarchii).
19. GetParentForGroup(string) – zwróć grupę rodzicielską dla grupy.
20. GetUserId() – zwróć identyfikator zalogowanego użytkownika.
21. GetUserName() – zwróć nazwę zalogowanego użytkownika.
22. GetUsersForGroup(string) – zwróć użytkowników dla grupy.
23. IsCurrentUserInGroup(string) – sprawdź, czy użytkownik należy do grupy.
24. IsGroupExists(string) – sprawdź, czy grupa istnieje.
25. IsUserInGroup(string, string) – sprawdź, czy użytkownik znajduje się w grupie.
26. RemoveCurrentUserFromGroup(string) – usuń zalogowanego użytkownika z grupy.
27. RemoveUserFromGroup(string, string) – usuń użytkownika z grupy.
28. RenameGroup(string, string) – zmień nazwę grupy.
29. UnAuthoriseCurrentUser() – nie autoryzuj zalogowanego użytkownika.
30. UnauthorisedUsers() – zwróć nieautoryzowanych użytkowników.
31. UnAuthoriseUser(string) – nie autoryzuj użytkownika.

3.2. Diagramy sekwencji oraz współpracy wybranych przypadków użycia

Na rys. 3 przedstawiono diagram współpracy[3] dla czynności – ***autoryzuj studenta***. Proces autoryzacji użytkownika *student* realizowany jest przez administratora po wybraniu przez niego odpowiedniej opcji na formularzu. Obiekt *formularz* uzyskuje listę grup poprzez wywołanie metody: *GetGroupList2()*, obiektu: *Database_Service* usługi sieciowej SZZD. Użytkownik musi być przypisany do grupy. Użytkownik, wybierając grupę, wymusza wysłanie przez formularz dwóch komunikatów. W odpowiedzi na pierwszy komunikat zostaje do formularza przesłana lista nieautoryzowanych studentów z wybranej grupy. Efektem drugiego komunikatu jest przesłanie do formularza listy nieautoryzowanych użytkowników z SRU. Następnie wysyłane są, przez *formularz autoryzacji studenta*, komunikaty do SRU oraz do SZZD efektem, których jest dodanie studenta do grupy – *studenci*.

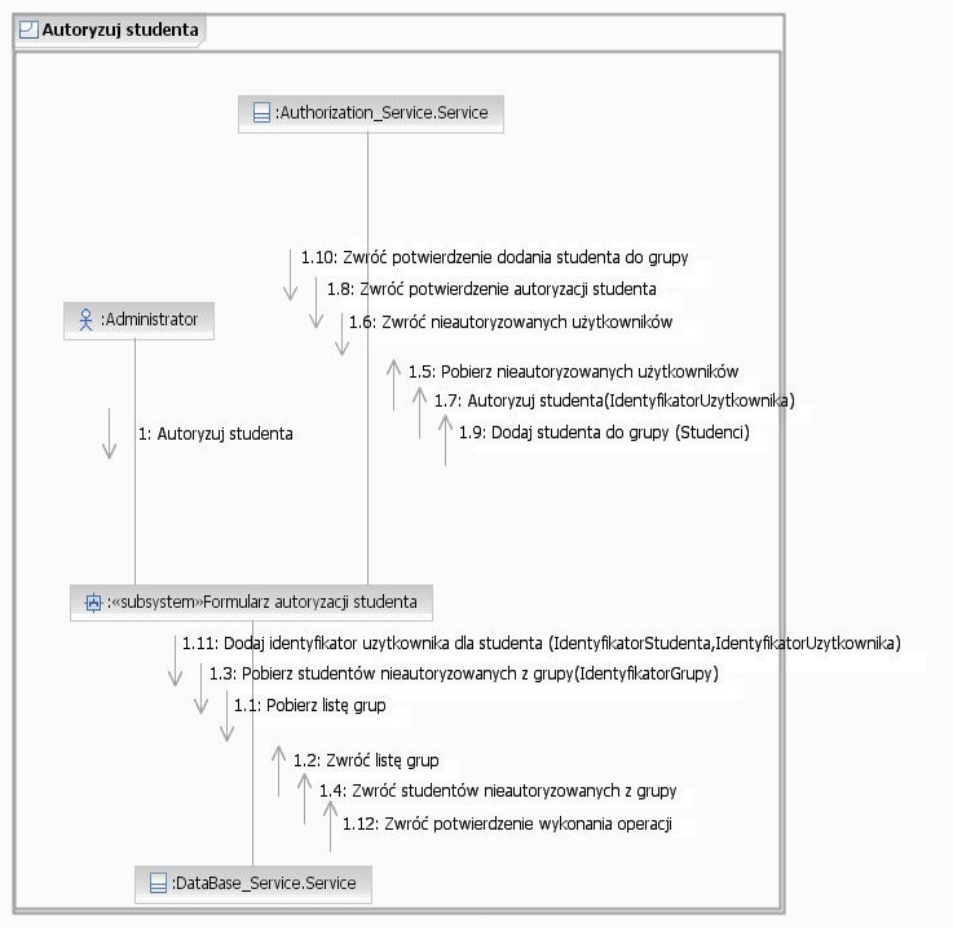
Na rys. 4 przedstawiono diagram sekwencji[4] dla czynności – ***dodaj studenta***.

[3] **Diagram współpracy** odwzorowuje powiązania pomiędzy obiektami, nie uwzględnia upływu czasu, służy do odwzorowywania efektów oddziaływania na pojedynczy obiekt [2].

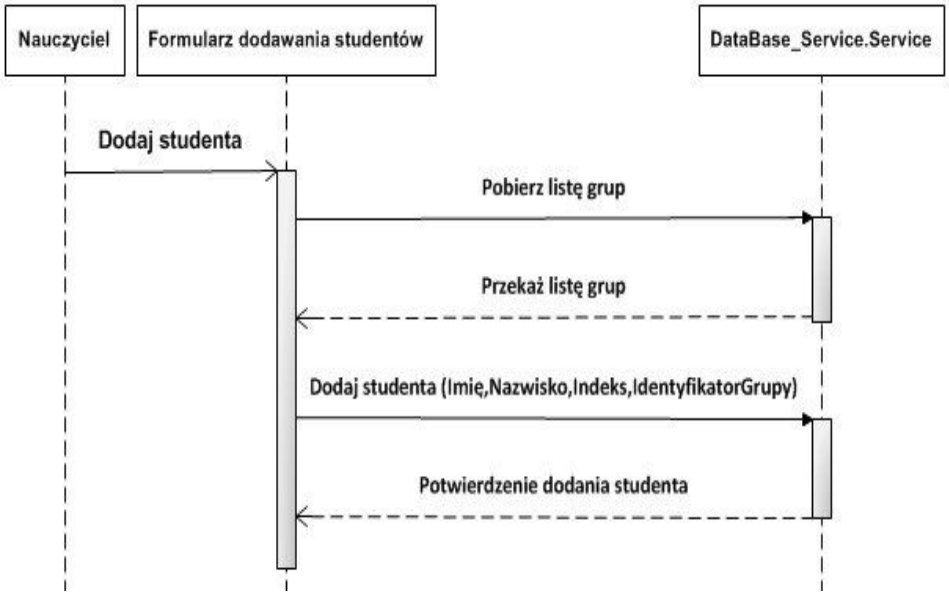
[4] **Diagram sekwencji** przedstawia komunikację obiektów z uwypukleniem czasu przesyłania komunikatów. Linia występująca pod każdym z obiektów odzwierciedla czas, a prostokąty aktywność obiektu [2].

Nauczyciel dodaje studenta poprzez wybranie opcji na formularzu. Obiekt formularz wywołuje metodę pobrania listy grup na obiekcie usługi sieciowej SZZD. Użytkownik wybiera grupę oraz dodaje informacje o studencie. Na podstawie tych informacji generowany jest komunikat dodania studenta do grupy. Obiekt usługi sieciowej przesyła zwrotnie potwierdzenie wykonania operacji.

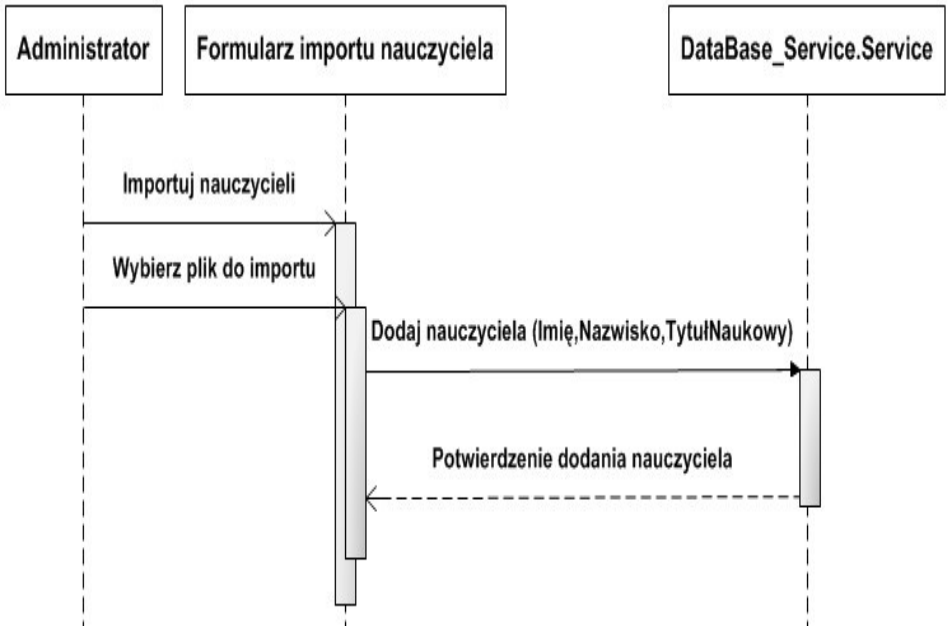
Na rys. 5 przedstawiono diagram sekwencji dla czynności – ***importuj nauczycieli***. Administrator importuje nauczycieli poprzez wybranie opcji na formularzu importu. Następnie wybierany jest plik, z którego będą importowane dane. Obiekt formularza wykonuje metodę dodania nauczyciela na obiekcie *DataBase_Service*.



Rys. 3. Diagram współpracy – autoryzuj studenta



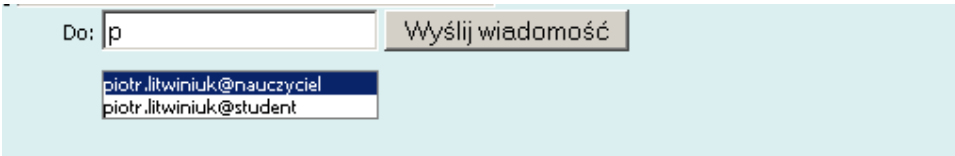
Rys. 4. Diagram sekwencji – dodaj studenta



Rys. 5. Diagram sekwencji – importuj nauczycieli

4. Implementacja projektu systemu

SWPN ma wbudowany mechanizm podpowiedzi zaimplementowany przy użyciu ASP.NET AJAX, którego przykład działania przedstawia rys. 6.



Rys. 6. Mechanizm podpowiedzi w SWPN

SWPN wykorzystuje metody usługi sieciowej SZZD do pobrania wszystkich adresów użytkowników. W systemie została zaimplementowana usługa sieciowa o nazwie: *AutoCompleteService*, która udostępnia metodę *GetAllMailAdresses*. Funkcja ta przyjmuje dwa argumenty, pierwszy argument określa ciąg znaków, który powinien się pokrywać z wyszukanym słowem, drugi argument określa maksymalną liczbę wyświetlanych słów. Lista wyświetlanych słów znajduje się w obiekcie – *suggestions* (tab. 1). Wyłuskanie wszystkich słów zawierających wzorzec jest wykonywane za pomocą polecenia SELECT na obiekcie *DataSet*. Implementację metody *GetAllMailAdresses* przedstawiono w tab. 1.

Tab. 1. Implementacja metody GetAllMailAdresses

```
public string[] GetAllMailAdresses(string prefixText,int count)
{
   Database_Service.Service ws = new Database_Service.Service();
   DataSet adresses = new DataSet();
   adresses = ws.GetAllMailAddresses();
   DataSet ds = new DataSet();
   DataTable dt = new DataTable();
   ds.Tables.Add(dt);
   dt.Columns.Add(new DataColumn("Word", typeof(string)));
   foreach (DataRow dr in adresses.Tables[0].Rows)
   {
      dt.Rows.Add(dr[0].ToString());
   }
   List<string> suggestions = new List<string>();
```

```
   DataRow[] words = ds.Tables[0].Select("Word LIKE '" +
   prefixText + "%'");
   int num = (words.Length < count) ? words.Length : count;
   for (int i = 0; i < num; i++)
   {
      suggestions.Add((string)words[i]["Word"]);
   }
   return suggestions.ToArray();
}
```

Aby móc wykorzystywać metody usług sieciowych w rozszerzeniach AJAX, należy za pomocą kontrolki *ScriptManager* zdefiniować lokalizację wykorzystanych usług. Istotne jest, aby *ScriptManager* znajdował się na każdym formularzu, na którym zostanie użyty AJAX. W tab. 2 przedstawiono implementację kontrolki *ScriptManager*.

Tab. 2. Implementacja kontrolki ScriptManager

```
<asp:ScriptManager ID="ScriptManager1" runat="server">
   <Services>
      <asp:ServiceReference Path="~/AutoCompleteService.asmx" />
   </Services>
</asp:ScriptManager>
```

Kontrolkę rozszerzenia *AutoCompleteExtender* wykorzystywano do wyświetlania podpowiedzi użytkownikowi. Właściwość *ServicePath* określa lokalizację usługi sieciowej zdefiniowaną w kontrolce *ScriptManager*. Atrybut *ServiceMethod* określa nazwę metody, wykorzystywaną do wyświetlania podpowiedzi. *MinimumPrefixLength* definiuje minimalną liczbę wpisanych znaków, która jest potrzebna do wyświetlania podpowiedzi, w SWPN jest to jeden znak. Właściwość *CompletionSetCount* określa maksymalną liczbę wyświetlanych podpowiedzi, przyjęto, że jest ich dwanaście. W tab. 3 przedstawiono implementację rozszerzenia *AutoComplete*.

Tab. 3. Implementacja rozszerzenia AutoComplete

```
<asp:TextBox ID="TextBox3" runat="server">
</asp:TextBox>
<cc1:AutoCompleteExtender ID="TextBox3_AutoCompleteExtender"
   runat="server" Enabled="True" TargetControlID="TextBox3"
   ServicePath="~/AutoCompleteService.asmx"
   ServiceMethod="GetAllMailAdresses" MinimumPrefixLength="1"
   CompletionSetCount="12" >
</cc1:AutoCompleteExtender>
```

5. Użytkowanie aplikacji

Na rys. 7 przedstawiono interfejs graficzny SWPN. Aplikacja zawiera dwa menu. Menu podręczne użytkownika (po lewej stronie) jest dla każdej zalogowanej osoby takie samo i dostęp do niego mają wszyscy, poza administratorem. Wydzielono takie pozycje, jak: *Strona główna*, *Profil*, *Wyślij wiadomość*, *Wyślij wiadomość do grupy*, *Wiadomości odebrane*, *Wiadomości wysłane* oraz *Wyloguj*. Pierwsza pozycja przenosi użytkownika na stronę startową, kolejna wyświetla dane personalne osoby przeglądającej zasoby aplikacji. W obrębie SWPN jest możliwa wymiana informacji (wiadomości) między aktorami korzystającymi z systemu. Wiadomość może być wysłana do jednej osoby bądź do grupy osób, które mają swoje konta w systemie. W zasobach o nazwach *Wiadomości odebrane* oraz *Wiadomości wysłane* znajdują się wiadomości odebrane lub wysłane przez zalogowanego użytkownika.

Po prawej stronie interfejsu użytkownik ma do dyspozycji osobisty notatnik, w którym może przechowywać informacje istotne dla niego w danym dniu. Standardowo w notatniku znajduje się czternaście pozycji, tyle, ile wynosi liczba godzin w ciągu dnia w WAT. Osoba może zdefiniować czynności do wykonania w wybranej godzinie, a SWPN prześle wartości do SZZD.

Menu główne aplikacji składa się z pozycji takich, jak: *Grupy*, *Studenci*, *Nauczyciele*, *Użytkownicy*, *Role*, *Przedmioty*, *Rodzaje zajęć*, *Zajęcia, Oceny*. Dostęp do tych pozycji jest zależny od tego, do jakiej grupy należy przeglądający zasoby użytkownik. Grupa w SWPN nazwana jest *rolą*, zatem pozycja *Role* odpowiada zarządzaniu grupami w SRU.

W SWPN została zdefiniowana rola – *nauczyciele*. Każdy nauczyciel może być jednocześnie w grupie *nauczyciele* oraz w grupie *administratorzy*. Wykładowca, jeśli ma przypisaną tylko rolę *nauczyciele*, ma dostęp do pozycji takich, jak: *Grupy*, *Studenci*, *Zajęcia*. Rola *nauczyciel* daje uprawnienia dostępu

do wszystkich pozycji, z wyjątkiem pozycji *Oceny* (taką sytuację przedstawia rys. 7). Dostęp do tej pozycji jest możliwy wówczas, gdy nauczyciel należy do grupy *administratorzy*. Istotna z punktu widzenia nauczyciela jest pozycja *Zajęcia*. Pozwala ona na uzyskanie dostępu przez użytkownika do informacji o prowadzonych przez niego zajęciach, umożliwiając: modyfikację aktywności grupy oraz studentów, eksport do plików, wystawianie ocen, zarządzanie skalą ocen. Pozycje *Grupy* oraz *Studenci* pozwalają wykładowcy na zarządzanie danymi odpowiadającymi grupom szkoleniowym oraz studentom.

Rys. 7. Interfejs graficzny wykorzystany w SWPN

Druga grupa użytkowników zdefiniowana w systemie to *administratorzy*. Użytkownicy tej grupy nie mają dostępu do: menu podręcznego, pozycji *Oceny* w menu głównym oraz pozycji dowiązanych do roli *nauczyciel*. Role w SWPN wykluczają się. Użytkownik przypisany do roli *nauczyciel* nie ma dostępu do pozycji menu dostępnych dla roli *administratora*, natomiast użytkownik przypisany do roli *administrator* nie ma dostępu do pozycji menu dostępnych dla roli *nauczyciel*. Użytkownik może być członkiem zarówno grupy *administratorzy*, jak również *nauczyciele*. Dopiero członkostwo w obydwu tych grupach daje pełne prawa do przeglądania zasobów aplikacji. Istotna z punktu widzenia realizacji przypisanych administratorowi funkcji jest pozycja w menu głównym – *Użytkownicy*. Opcja ta pozwala na realizację, przez *administratora*, procesu autoryzacji użytkowników: *nauczyciel* oraz *student*. Proces autoryzacji polega na powiązaniu osoby mającej konto w SRU z użytkownikiem z SZZD. Aby mieć dostęp do aplikacji, konto każdego aktora musi zostać „zaakceptowane” przez osobę upoważnioną.

Użytkownik przypisany do roli *studenci* ma dostęp do pozycji *Oceny* (tylko do odczytu). Inne pozycje menu głównego są niedostępne dla użytkownika tej grupy. Osoba należąca do grupy *administratorzy* może zezwolić na dostęp studentowi do zasobów administracyjnych. Wykonuje się to za pomocą pozycji menu – *Role→Przypisz rolę administratora*. Istnieje możliwość usunięcia takiego powiązania poprzez wybranie pozycji menu – *Role→Usuń przypisanie administratora*.

Podsumowanie

Aplikacja wspomagająca pracę nauczyciela znacznie ułatwia proces zdalnego zarządzania wynikami uzyskiwanymi przez studentów. Jednym z założeń przyjętych przy tworzeniu projektu SWPN było umożliwienie korzystania z niej zarówno nauczycielom, jak i studentom. Dane przechowywane przez SZZD mogą być importowane i eksportowane do pliku za pomocą aplikacji „dziennik nauczyciela".

Jednym z powodów użycia usług sieciowych było zapewnienie naturalnego podziału projektu na moduły. Wykorzystanie usług sieciowych pozwala na rozbudowę SWPN oraz łatwą, bez konieczności zmian kodu SWPN, modyfikację istniejących funkcji.

Jednym z problemów rozwiązanych podczas budowy aplikacji było zintegrowanie danych w systemach SZZD oraz SRU. Kolejny problem stanowiło maksymalne wykorzystanie możliwości usług sieciowych. Przesyłanie danych poprzez sieć Internet wymagało, aby SWPN wykorzystywał mechanizmy transmisji zapewniające odpowiedni poziom bezpieczeństwa. W obecnym rozwiązaniu do zabezpieczenia transmisji wykorzystano protokół SSL. Wykorzystywany jest on przy komunikacji z systemem uwierzytelniania – SRU, zapewniając bezpieczne przesyłanie informacji uwierzytelniających. Istotne, z punktu widzenia bezpieczeństwa było również zapewnienie odpowiedniego mechanizmu walidacji danych wprowadzanych przez użytkowników. Do realizacji procesu walidacji danych w SWPN wykorzystano kontrolki – *walidatorów*. Komunikacja z użytkownikami w systemie odbywa się poprzez wewnętrzny komunikator przystosowany do wymiany informacji, tylko pomiędzy zarejestrowanymi osobami. Zadaniem wykładowcy jest wybór sposobu oceniania studentów oraz skali ocen, co w intuicyjny sposób zostało zaimplementowane w SWPN. Nauczyciel w dowolnym momencie może eksportować wyniki studentów, przechowywane przez SZZD, do pliku „pdf" oraz „xls". Wymiana informacji pomiędzy aplikacją SWPN a systemem „e-dziekanat", w którym są przechowywane nazwy przedmiotów, dane personalne studentów oraz wykładowców, odbywa się za pomocą plików „txt" oraz „xls". SWPN został tak zaprojektowany, żeby istniała możliwość wdrożenia go w sieciach lokalnych na innych uczelniach niż WAT.

System może być rozwijany poprzez dodanie: możliwości wymiany plików pomiędzy użytkownikami systemu, modułu zdalnego egzaminowania, modułu zarządzania obciążeniem sal laboratoryjnych, czy też modułu zarządzania oprogramowaniem wymaganym do realizacji ćwiczeń laboratoryjnych.

Literatura

[1] CONOLLY R., ASP.NET 2.0. *Projektowanie aplikacji internetowych*, Helion, Gliwice, 2008.

[2] GRADY B., RUMBAUGH J., JACOBSON I., *UML przewodnik użytkownika*, WNT, Warszawa, 2002.

[3] KANJILAL J., PUTREVU S., *ASP.NET Ajax*, Helion, Gliwice, 2009.

[4] KWIATEK P., *System rejestracji użytkowników w serwisie internetowym*, Praca dyplomowa, Wydział Cybernetyki WAT, 2010.

[5] LIBERTY J., HURWITZ D., *ASP.NET. Programowanie*, Helion, Gliwice, 2007.

[6] LITWINIUK P., *Aplikacja ASP.NET wspomagająca pracę nauczyciela*, Praca dyplomowa, Wydział Cybernetyki WAT, 2010.

[7] POLKOWSKI S., *System zdalnego zarządzania współdzielonymi danymi oparty o usługi WWW*, Praca dyplomowa, Wydział Cybernetyki WAT, 2010.

A teacher assistant system

ABSTRACT**:** In this paper a project and implementation of a teacher-assistant software application system is presented. Requirements of the system have been defined. Selected interaction diagrams are given in the UML notation. An issue of cooperation with outside systems via web services is presented too. Selected aspects of the system implementation are discussed.

KEY WORDS: Web application, Web Services, UML, ASP.NET, AJAX ASP.NET.

Praca wpłynęła do redakcji: 14.09.2010