System Zdalnego Egzaminowania

Michał PIOTROWICZ, Jan CHUDZIKIEWICZ

Instytut Teleinformatyki i Automatyki WAT,
ul. Gen. S. Kaliskiego 2, 00-908 Warszawa
michal.piotrowicz@op.pl, jchudzikiewicz@wat.edu.pl

STRESZCZENIE: W artykule przedstawiono opis projektu oraz implementacji wybranych elementów Systemu Zdalnego Egzaminowania. Zaprezentowano mechanizmy obsługi bazy danych z wykorzystaniem procedur składowanych. Omówiono sposób integracji opracowywanego systemu z istniejącym systemem dziennika nauczyciela oraz sposób użytkowania aplikacji.

SŁOWA KLUCZOWE: aplikacja internetowa, usługa sieciowa, weryfikacja wiedzy studenta, procedury składowane, UML, ASP.NET.

1. Wstęp

Weryfikacja wiedzy ucznia, czy też studenta, może przyjmować wiele form. Wybór formy przeprowadzenia tego procesu uwarunkowany jest rożnymi czynnikami, do których między innymi możemy zaliczyć np. liczbę studentów podlegających temu procesowi czy też zakres materiału, którego ten proces ma dotyczyć. Jedną z form, jaką może przyjąć ten proces, jest „test". Zgodnie z definicją przedstawioną w słowniku języka polskiego PWN, test jest to: *badanie eksperymentalne lub narzędzie pomiaru składające się z zestawu standaryzowanych zadań, pytań*. Będąc w zgodności z przedstawioną definicją, postanowiono opracować narzędzie pozwalające na weryfikację wiedzy studenta. Narzędzie to przyjęło formę aplikacji rozproszonej [5], [7] i zostało włączone do opracowanego w 2010 roku w Instytucie Teleinformatyki i Automatyki elektronicznego dziennika nauczyciela.

W skład elektronicznego dziennika nauczyciela wchodzą trzy elementy:

- System Rejestracji Użytkowników (SRU) [2];
- System Zdalnego Zarządzania Danymi (SZZD) [3];

– System Wspomagania Pracy Nauczyciela (SWPN) [4].

Schemat poglądowy umiejscowienia ***Systemu Zdalnego Egzaminowania*** (SZE) [1] w elektronicznym dzienniku nauczyciela przedstawiono na rys. 1.

Rys. 1. Schemat powiązań pomiędzy aplikacjami

Celem budowy SZE było rozszerzenie funkcjonalności elektronicznego dziennika nauczyciela o możliwość zdalnej weryfikacji wiedzy studentów. System SZE jest udostępniony jako opcja SWPN. Przyjęcie takiego rozwiązania nieznacznie zwiększyło złożoność obsługi SZE z punktu widzenia użytkownika, uprościło natomiast sposób realizacji procesu uwierzytelniania, bo użytkownik uwierzytelniany jest na poziomie SWPN. SZE przejmuje żeton uwierzytelnienia użytkownika [5], [7] od SWPN. Projektując SZE, przyjęto założenie polegające na wbudowaniu w SZE zarówno funkcji edycyjnej (tworzenie oraz modyfikacja testów), jak i funkcji przeprowadzania testów. Funkcja przeprowadzania testów polega na zdalnym udostępnianiu dla określonej grupy użytkowników (np. studentów) testów w celu weryfikacji ich wiedzy. Wynik (ocena) wykonania testu jest automatycznie dołączany do innych wyników, które użytkownik dotychczas uzyskał. Wszystkie uzyskane przez użytkownika wyniki są dostępne z poziomu SWPN.

2. Wymagania funkcjonalne

Wymagania funkcjonalne zostały określone na podstawie wiedzy zdobytej przez współautora artykułu na przestrzeni kilkunastu lat pracy dydaktycznej, jak również przeglądu dostępnych na rynku aplikacji wspomagających proces egzaminowania. Pozwoliło to na określenie trzech głównych grup funkcji, którymi powinny charakteryzować się tego typu systemy:

- zarządzanie pytaniami oraz odpowiedziami;
- zarządzanie egzaminami;
- realizacja procesu egzaminowania.

Tworząc test, należy określić jego strukturę. Jednym z elementów tej struktury są pytania oraz przypisane im odpowiedzi. Struktura opisuje ponadto, między innymi takie elementy jak: czas realizacji testu, zakresy pytań (grupowanie pytań w obszarach tematycznych), typ testu (jednokrotnego lub wielokrotnego wyboru). Jak już wcześniej wspomniano, zaprezentowane rozwiązanie łączy ze sobą funkcję edytora testów oraz systemu przeprowadzania testów. Dodanie funkcji edytora miało na celu ułatwia obsługi systemu przez użytkownika (nie musi on uruchamiać oddzielnego modułu, może na bieżąco modyfikować zestawy testów). Proces budowy nowego testu, poza określeniem parametrów dodatkowych, wymaga dodawania nowych pytań. Tworzenie pytań realizowane jest przez użytkownika poprzez wypełnienie formularza WWW, który pozwala na wprowadzenie:

a. Treści pytania.
b. ***Przedmiotu*** – przedmiot, z którego przeprowadzany będzie egzamin.
c. ***Grupy tematycznej*** – pozwalającej na zawężenie zakresu materiału w ramach przedmiotu.
d. ***Załącznika*** – możliwe jest wprowadzenie załącznika w formie rysunku.

Do każdego pytania użytkownik ma możliwość wprowadzenia dowolnej liczby odpowiedzi. Określenie odpowiedzi wymaga wprowadzenia następujących danych:

a. Treści odpowiedzi.
b. ***Poprawności odpowiedzi*** – jest to wartość logiczna pozwalająca na określenie, czy dana odpowiedź jest poprawna (prawdziwa), czy też nie (fałszywa).
c. ***Wartości punktowej odpowiedzi*** – określa liczbę punktów, jaka zostanie przyznana użytkownikowi za poprawną odpowiedź. Jeżeli użytkownik nie odpowie poprawnie to, określona w tym polu liczba punktów zostanie odjęta od ogólnej liczby zdobytych przez użytkownika punktów. Przyjęte rozwiązanie pozwala na różnicowanie trudności pytań.

Użytkownik budujący test ma możliwość określenia typu testu: jednokrotnego lub wielokrotnego wyboru. Pole „*Wartość punktowa odpowiedzi”* w głównej mierze znajduje zastosowanie w tym drugim przypadku.

W celu zapewnienia nauczycielowi jak największej swobody podczas tworzenia nowych testów, system pozwala na zdefiniowanie wielu parametrów konfiguracyjnych. Takie rozwiązanie umożliwia przeprowadzanie w systemie mniej złożonych *aktywności*, takich jak kartkówki oraz kolokwia, jak również

tych bardziej rozbudowanych, takich jak zaliczenia i egzaminy. Dodanie nowego egzaminu wiąże się z koniecznością podania następujących parametrów:

a. Nazwy oraz opisu egzaminu.
b. Przedmiotu oraz typu zajęć, z którego jest dany egzamin.
c. Grupy, dla której przeznaczony jest dany egzamin.
d. Czasu trwania egzaminu oraz ilości pytań.
e. Liczby odpowiedzi poprawnych (w przypadku testu wielokrotnego wyboru) oraz niepoprawnych.
f. Liczby poprawek, które student może wykorzystać.
g. Typu testu – test jednokrotnego lub wielokrotnego wyboru.
h. Rozkładu pytań z grup tematycznych, które będą losowane do egzaminów. Pozwala na określenie liczby pytań losowanych do testu z każdej ze zdefiniowanych grup tematycznych.

Po określeniu struktury testu zostaje on dopisany do puli wszystkich testów, znajdujących się w bazie danych. Użytkownik, który jest Autorem testu, przed udostępnieniem go studentom, może zweryfikować jego poprawność poprzez próbne wykonanie testu. Wszelkie parametry testu, jak również wartości określone dla poszczególnych pytań oraz odpowiedzi, są w pełni modyfikowalne przez Autora w dowolnym momencie.

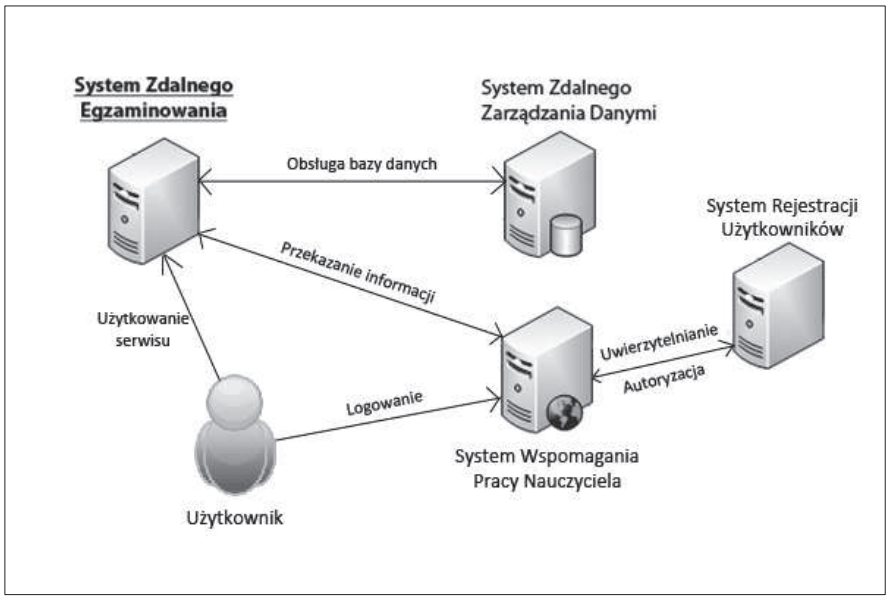
3. Realizacja systemu oraz jego integracja z SWPN

SZE jest oprogramowaniem rozszerzającym funkcjonalność SWPN i pomimo iż został zaprojektowany jako oddzielna aplikacja, w pełni integruje się z pozostałymi elementami SWPN. Projekt SZE opiera się w pełni na klasycznym trzy-warstwowym modelu architektury, składającym się z:

- warstwy prezentacji;
- warstwy logiki biznesowej;
- warstwy danych.

Dostęp do SZE jest przydzielony każdemu użytkownikowi, który ma konto w SWPN. Uprawnienia do SZE są zależne od statusu, jaki został przydzielony użytkownikowi przez administratora SWPN. Jeżeli użytkownikowi został przydzielony status *nauczyciel*, to ma on uprawnienia do budowy i modyfikacji testów, jak również udostępniania testów dla określonej grupy użytkowników. Użytkownik o statusie *student* otrzymuje uprawnienie do wykonania udostępnionego dla jego grupy testu oraz wglądu w dotychczas uzyskane wyniki (z poziomu SWPN). Na rys. 2 przedstawiono umiejscowienie SZE w systemie dziennika nauczyciela oraz sposób współdziałania poszcze-

gólnych komponentów systemu. Użytkownik, chcąc uzyskać dostęp do SZE, loguje się w SWPN. Z poziomu panelu SWPN, poprzez wybór odpowiedniej opcji menu, może uzyskać dostęp do SZE. Przydzielony mu żeton dostępu przekazywany jest pomiędzy SWPN a SZE. Natomiast modyfikacja testów (użytkownik – *nauczyciel*) realizowana jest bezpośrednio z poziomu SZE poprzez SZZD. W operacji tej nie pośredniczy SWPN.



Rys. 2. Schemat współdziałania aplikacji

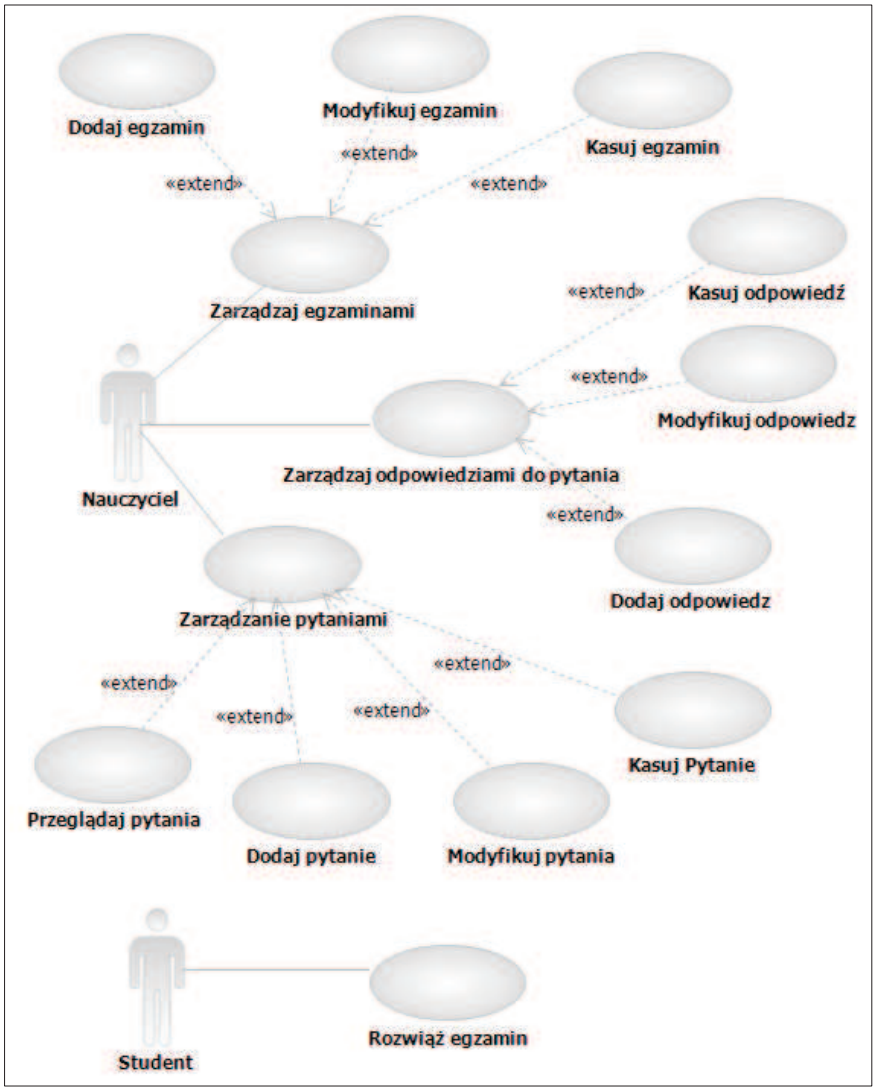
3.1. Realizacja Systemu Zdalnego Egzaminowania

3.1.1. Projekt wybranych elementów systemu

Istotne, z punktu widzenia realizacji projektu, jest określenie użytkowników, którzy będą z systemu korzystać, oraz relacji pomiędzy nimi a elementami systemu. Opisuje się to np. za pomocą diagramu przypadków użycia, który dla projektowanego systemu został przedstawiony na rys. 3. W ramach SZE przewiduje się dwie grupy użytkowników: nauczyciele oraz studenci, które to grupy na przedstawionym diagramie są aktorami: nauczyciel i student [6]. Aktor *nauczyciel* komunikuje się z trzema modułami SZE reprezentowanymi przez przypadki użycia, do których zaliczamy: *Zarządzaj*

egzaminami, *Zarządzaj pytaniami*, *Zarządzaj odpowiedziami do pytania.*

Dla omówienia sposobu zapisu danych do bazy danych opisano przypadek użycia – *Dodaj pytanie* łącznie z procedurą składowaną [5], [7] oraz funkcją ją wykorzystującą. Dla omówienia odczytu danych przedstawiono procedurę składowaną oraz metodę pobierania *aktywności* [4].

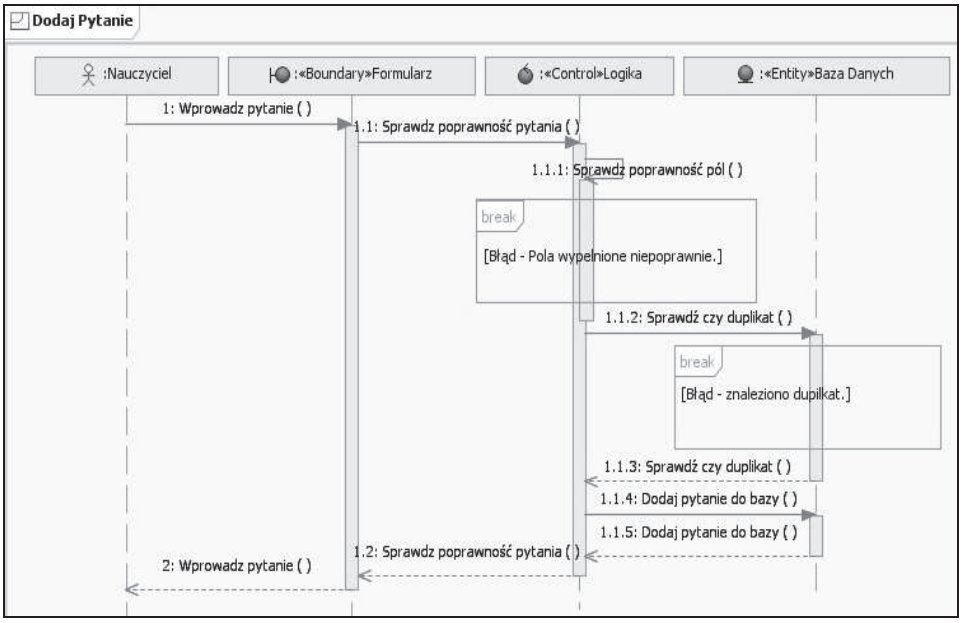


Rys. 3. Diagram przypadków użycia SZE

Zarządzanie pytaniami rozszerzane jest przez następujące przypadki użycia:

- *dodawanie pytań*;
- *przeglądanie pytań*;
- *edycja pytań*;
- *kasowanie pytań*.

Na rys. 4 przedstawiono realizację przypadku użycia – *Dodaj pytanie* – opisaną przy użyciu diagramu sekwencji. Diagram przedstawia sekwencję przejść pomiędzy użytkownikiem, formularzem, logiką aplikacji oraz bazą danych. Proces dodawania pytania realizowany jest w trzech krokach. W pierwszym sprawdzana jest poprawność wszystkich wprowadzonych danych. W kolejnym następuje weryfikacja zawartości bazy danych pod kątem wystąpienia duplikatu danego pytania. Poprawne zakończenie dwóch pierwszych kroków powoduje zapisanie pytania w bazie danych, co stanowi ostatni krok realizacji procesu dodawania pytania do bazy danych.



Rys. 4. Diagram sekwencji dla przypadku użycia *Dodaj pytanie*

3.1.2. Implementacja wybranych elementów systemu

Proces obsługi zapisu pytań do bazy danych zrealizowano poprzez procedurę składowaną [5], [7], którą przedstawiono na rys. 5. Parametrami wej-

ściowymi procedury o nazwie *szeDodajPytanie* są:

a. ***@Nazwa_Przedmiotu*** – nazwa przedmiotu, typ *varchar* (łańcuch znaków) o długości do 255 znaków.
b. ***@ID_Tematu*** – identyfikator tematu, typ *int* (liczba całkowita).
c. ***@Tresc_Pytania*** – treść pytania, typ *text* (łańcuch znaków) o długości do 8000 znaków.
d. ***@Nazwa_obrazka*** – nazwa obrazka dodanego w formie załącznika, typ *varchar* (łańcuch znaków) o długości do 255 znaków.

Dla każdego z zadeklarowanych parametrów, po wskazaniu typu, określona została jego wartość domyślna.

```
CREATE PROCEDURE [dbo].[szeDodajPytanie]
      @Nazwa_Przedmiotu varchar(255) = '',
      @ID_Tematu int = 0,
      @Tresc_pytania text = '',
      @Nazwa_obrazka varchar(255) = ''
AS
BEGIN
      SET NOCOUNT ON;
      INSERT INTO dbo.Pytanie
      (
            Nazwa_przedmiotu,
            ID_Tematu,
            Tresc_Pytania,
            Nazwa_obrazka
      )
      VALUES
      (
            @Nazwa_Przedmiotu,
            @ID_Tematu,
            @Tresc_pytania,
            @Nazwa_obrazka
      )
      RETURN @@Identity
END
GO
```

Rys. 5. Deklaracja procedury składowanej

Procedura rozpoczyna się po słowie kluczowym *AS* i oparta jest na instrukcji *Insert*. Składnia tego polecenia wymaga podania nazwy tabeli, do której zostanie dodany rekord, nazw kolumn oraz nazw parametrów. Instrukcja *Return*, *w* przypadku poprawnego wykonania polecenia *Insert*, zwraca wartość identyfikatora pytania, natomiast w przypadku błędu zwraca wartość ujemną. Procedury edytujące oraz kasujące rekordy bazują odpowiednio na instrukcji *Update* i *Delete*, oraz na klauzuli *Where*, która umożliwia dopasowanie

konkretnych rekordów.

Na rys. 6 przedstawiono kod metody o nazwie *AddQuestion*, która wywołuje procedurę *szeDodajPytanie* w celu zapisania pytania do bazy danych. Działanie metody możemy podzielić na cztery etapy:

- otwarcie połączenia;
- zadeklarowanie parametrów wejściowych oraz wyjściowych;
- wykonanie zapytania i odbiór wyniku;
- zamknięcie połączenia.

```
public static Int32 AddQuestion(string Nazwa_przedmiotu,
  int ID_Tematu,string Tresc_pytania,string Nazwa_obrazka)
{
   // Otwarcie połączenia
  SqlConnection conn = new SqlConnection();
  conn.ConnectionString = GlobalVars.ConnectionString;
  conn.Open();
   // Zadeklarowanie parametrów wejściowych oraz wyjściowych
  SqlCommand command = new SqlCommand();
  command.Connection = conn;
  command.CommandType = CommandType.StoredProcedure;
  command.CommandText = "szeDodajPytanie";
  command.Parameters.AddWithValue
          ("@Nazwa_Przedmiotu", Nazwa_przedmiotu);
  command.Parameters.AddWithValue
          ("@ID_tematu", ID_Tematu);
  command.Parameters.AddWithValue
          ("@Tresc_pytania", Tresc_pytania);
  command.Parameters.AddWithValue
          ("@Nazwa_obrazka", Nazwa_obrazka);
  SqlParameter retParam =
      new SqlParameter("@RETURN_VALUE", SqlDbType.Int);
  retParam.Direction = ParameterDirection.ReturnValue;
  command.Parameters.Add(retParam);
   // Wykonanie zapytania i odbiór wyniku
  SqlDataReader reader = command.ExecuteReader();
  reader.Close();
  Int32 result = Convert.ToInt32(retParam.Value);
   // Zamknięcie połączenia
  conn.Close();
  if (result == 0) result = 1;
  return result;
}
```

Rys. 6. Definicja metody dodającej pytanie do bazy danych

Otwarcie połączenia realizowane jest przez wywołanie metody *Open* klasy *SqlConnection*. Parametrem metody *Open* jest wartość *ConnectionString*, zawierająca parametry takie jak: adres bazy danych, opis połączenia z bazą

danych oraz informacje umożliwiające zalogowanie do bazy danych. Zadeklarowanie parametrów wejściowych oraz wyjściowych polega na utworzeniu obiektu klasy *SqlCommand*. Jako atrybuty tego obiektu podawane są: utworzony wcześniej obiekt ciągu połączenia, typ oraz nazwa procedury składowanej. Określany jest również parametr zwrotny zapytania (w przypadku procedur składowych jest to wartość typu *int*) oraz parametry przekazywane w trakcie wywołania procedury. Wykonanie procedury składowej zrealizowane jest poprzez wywołanie funkcji *ExecuteReader* klasy *SqlDataReader*. Zwrotnie przekazana zostaje wartość *Result*, informująca o poprawności wykonania operacji. Zakończenie dodawania pytania do bazy danych polega na zamknięciu połączenia przy użyciu funkcji *Close* oraz zwróceniu wartości uzyskanej z bazy danych.

Odczyt danych również bazuje na procedurach składowanych. Dla zobrazowania odczytu danych z bazy danych omówiono mechanizm pobierania *aktywności*. Do pobierania danych z tabel służy polecenie *Select*, które standardowo umożliwia wyświetlenie wyników tylko z jednej tabeli. Bardzo często zdarza się, że praktyczniej byłoby uzyskać dane powiązane ze sobą z wielu różnych tabel. Na rys. 7 przedstawiono rozwiązanie wykorzystujące polecenie *Select* zawarte w procedurze składowanej. Procedura ta umożliwia pobranie wszystkich *aktywności* z bazy na podstawie nazwy oraz informacji dotyczących zajęć.

```
CREATE PROCEDURE [dbo].[szeListaAktywnosci]
      @Nazwa varchar(255),
      @Id_rodzaju_zajec int = 0,
      @Id_przedmiotu int = 0
AS
BEGIN
      SET NOCOUNT ON;
      SELECT * FROM dbo.Aktywnosci AS a
      Inner Join dbo.Zajecia as z
      On a.Id_zajec = z.Id_zajec
      WHERE
      z.Id_rodzaju_zajec = @Id_rodzaju_zajec AND
      z.Id_przedmiotu = @Id_przedmiotu AND
      a.Nazwa_aktywnosci = @Nazwa
END
```

Rys. 7. Pobieranie listy *aktywności*

Słowem kluczowym jest w tym przypadku polecenie *Inner Join*, które pozwala na połączenie wyników z różnych tabel zgodnie z podanym warunkiem. W tym przypadku rekordy tabeli *Zajęcia* zostaną dołączone do rekordów tabeli *Aktywności*, zgodnie z identyfikatorem zajęć (warunek umieszczony po słowie *On*). Zestawione w ten sposób rekordy można poddać

filtracji przy użyciu polecenia *Where*, sprawdzając pola zarówno tabeli *Zajęcia* (nazwy pól poprzedzone literą „z") jak i tabeli *Aktywności* (nazwy pól poprzedzone literą „a"). Zaprezentowana na rys. 7 procedura nie ma instrukcji *Return*. W przypadku gdy procedura zostanie wykonana poprawnie, będzie zwrócona domyślna wartość równa 0. Jeżeli procedura nie wykona się poprawnie, zwróci wartość ujemną.

Realizacja metody pobierającej dane z bazy danych została przedstawiona na rys. 8. W przypadku tej metody inaczej niż w przypadku metody *AddQuestion* realizowane jest wywołanie procedury składowanej. Różnica (poza oczywiście zestawem parametrów) polega na użyciu zamiast klasy *SqlDataReader* klasy *SqlDataAdapter*. Klasa ta zawiera funkcję *Fill*, umożliwiającą wypełnienie obiektu *DataSet* rekordami, które zostały pobrane przez wywołaną procedurę składową. Obiekt *DataSet* jest kontenerem danych umożliwiającym przechowywanie tabel z bazy danych w pamięci [7]. W przeciwieństwie do metody *AddQuestion* metoda *GetActiviesIDByNameTypeOfLessonAndSubject*, nie wykorzystuje parametru zwrotnego *ReturnValue* – wynikiem działania metody jest jedynie obiekt *DataSet* zwracany przez usługę sieciową.

```
public static DataSet GetActiviesIDByNameTypeOfLessonAndSubject
       (string nazwa, int id_przedmiotu, int Id_rodzaju_zajec)
{
    // Otwarcie połączenia
    SqlConnection conn = new SqlConnection();
    conn.ConnectionString = GlobalVars.ConnectionString;
    conn.Open();
    // Zadeklarowanie parametrów wejściowych oraz wyjściowych
    SqlCommand command = new SqlCommand();
    command.Connection = conn;
    command.CommandType = CommandType.StoredProcedure;
    command.CommandText = "szeListaAktywnosci";
    command.Parameters.AddWithValue
                    ("@Nazwa", nazwa);
    command.Parameters.AddWithValue
                    ("@Id_rodzaju_zajec", Id_rodzaju_zajec);
    command.Parameters.AddWithValue
                    ("@Id_przedmiotu", id_przedmiotu);
    // Wykonanie zapytania i odbiór wyniku
    DataSet ds = new DataSet("ListaAktywnosci");
    SqlDataAdapter da = new SqlDataAdapter(command);
    da.Fill(ds);
    conn.Close();
    return ds;
}
```

Rys. 8. Definicja metody pobierającej rekordy z bazy danych

3.2. Integracja z pozostałymi systemami

Podstawowym założeniem utworzonego systemu było rozszerzenie funkcjonalności istniejącej już aplikacji, a nie tworzenie niezależnego narzędzia. Do istniejącego systemu SWPN dodano funkcje dostępne dla użytkownika z poziomu menu SWPN. Zestaw dostępnych funkcji zależny jest od roli, do jakiej został przypisany aktualnie zalogowany użytkownik. Użytkownik o roli *nauczyciel* ma dostępne następujące opcje (funkcje) menu:

- dodawanie pytań, oraz zarządzanie pytaniami;
- dodawanie odpowiedzi oraz zarządzanie odpowiedziami;
- dodawanie egzaminów oraz zarządzanie egzaminami;
- podgląd rozwiązywanych egzaminów.

Użytkownik o roli *student* ma dostępną opcję *rozwiązywanie egzaminu*.

Dla zapewnienia działania SZE rozszerzono funkcje obsługi bazy danych systemu SZZD. Rozszerzenie sprowadziło się do dodania nowych tabel, procedur składowanych oraz metod usługi sieciowej, które to metody umożliwiają dodawanie, modyfikowanie oraz kasowanie danych. Wszelkie dane związane z SZE są umieszczone w następujących tabelach:

a. ***Pytania*** – tabela zawiera treść pytania, jego identyfikator, przedmiot, z którego jest pytanie, moduł, nazwę załącznika (obrazek).
b. ***Odpowiedzi*** – tabela zawiera treść odpowiedzi, jej identyfikator, wartość logiczną (prawda/fałsz), wartość punktową oraz identyfikator pytania.
c. ***Egzaminy*** – tabela zawiera podstawowe informacje na temat szkieletu egzaminu, takie jak: przedmiot, liczba pytań, liczba poprawek, format.
d. ***Testy*** – tabela zawiera utworzone dla każdego użytkownika testy.
e. ***Aktywne testy*** – tabela zawiera informacje na temat rozwiązywanych testów.
f. ***Moduły*** – tabela zawiera nazwy grup tematycznych.

Do obsługi bazy danych, jak już wspomniano wcześniej, wykorzystano procedury składowane. Wybór takiego rozwiązania podyktowany był, między innymi, uproszczeniem sposobu realizacji odwołań do bazy danych, jak również podniesieniem poziomu bezpieczeństwa realizacji tych odwołań. Przy projektowaniu tego typu rozwiązań należy mieć na uwadze potencjalne źródła zagrożeń, których przykładem jest atak typu *SQL Injection*. Procedura składowana to przechowywany po stronie serwera i kompilowany przy pierwszym uruchomieniu zbiór zapytań SQL, który może być wielokrotnie wywoływany. Ponieważ są to zamknięte bloki kodu T-SQL, użytkownik, aby wprowadzać własne wartości do zapytania, musi wykorzystać zdefiniowane w procedurze parametry. Restrykcyjne stosowanie typowania parametrów

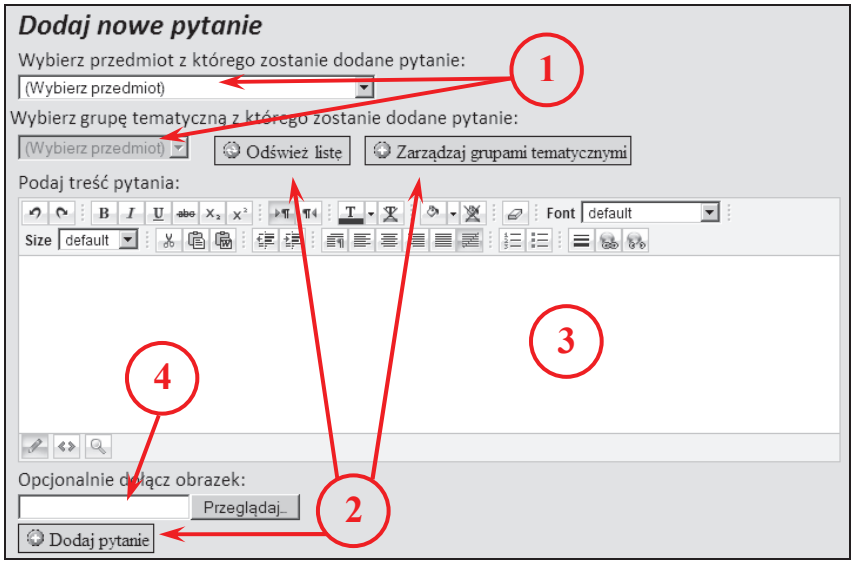
pozwoli na ograniczenie możliwych błędów, związanych z wprowadzeniem wartości o niepoprawnym formacie. Wywołana wraz z parametrami procedura przetwarza kod SQL, a następnie zwraca ewentualne rekordy oraz wartość *Result*, która informuje o poprawności wykonania kodu (zwracana wartość jest typu *Integer*).

4. Użytkowanie aplikacji

Aplikacja wspomagająca proces egzaminowania oraz dodatkowe metody usługi sieciowej [5], [7] zostały wykonane z wykorzystaniem .NET Framework oraz języka C#.

SZE umożliwia dodawanie własnych pytań, odpowiedzi i testów, a także edytowanie ich. Oprócz zarządzania danymi, aplikacja pozwala na przeprowadzenie testu dla określonej grupy studentów.

Na rys. 9 przedstawiono formularz umożliwiający przeprowadzenie procesu edycji już istniejącego pytania lub dodania nowego. Do budowy tego (i innych występujących w aplikacji) formularza wykorzystano obsługiwane w ASP.NET mechanizmy, takie jak: *Master Page*, *CSS*, *AJAX*, *UserControl* [7], [8], [9].



Rys. 9. Formularz dodawania (edycji) pytania

Proces wprowadzania oraz modyfikacji danych opiera swoje działanie na standardowych kontrolkach serwera WWW [5], [7], takich jak: *TextBox*, *Button* oraz *DropDownList*. Kontrolki te zostały zgrupowane w jednej kontrolce użytkownika (obiekt *UserControl*), która pełni rolę kontenera dla tych elementarnych obiektów. Kod opisujący kontrolkę użytkownika obsługującą modyfikację pytania został przedstawiony na rys. 10.

```
<%    @Register    src="AddQuestion.ascx"    tagname="AddQuestion"
tagprefix="uc1" %>
   <asp:View ID="View2" runat="server">
   <uc1:AddQuestion ID="AddQuestion1" runat="server" />
</asp:View>
```

Rys. 10. Definicja kontrolki użytkownika obsługującej modyfikację pytań

Użyty do budowy kontrolki obiekt *AddQuestion1* składa się z następujących elementów:

a. Dwóch kontrolek *DropDownList* – pierwsza z nich wyświetla listę dostępnych dla nauczyciela przedmiotów, druga (aktywowana po wybraniu przedmiotu) wyświetla listę grup tematycznych wskazanego przedmiotu (na rys. 9 oznaczone 1).
b. Trzech kontrolek *LinkButton* – pierwsza z nich umożliwia zarządzanie tematami, druga powoduje odświeżanie listy grup tematycznych, trzecia natomiast realizuje proces dodawania pytania (na rys. 9 oznaczone 2).
c. Kontrolki *HTMLEditor* – obiekt pakietu *AJAX Control Toolkit* umożliwiający wprowadzanie oraz formatowanie tekstu (na rys. 9 oznaczone 3).
d. Kontrolki *FileUpload* – umożliwia załączenie pliku graficznego (na rys. 9 oznaczone 4).
e. Kontenera *UpdatePanel* – pozwala na asynchroniczne odświeżanie listy grup tematycznych (element niewidoczny dla użytkownika, wykorzystywany do odświeżania fragmentu strony).

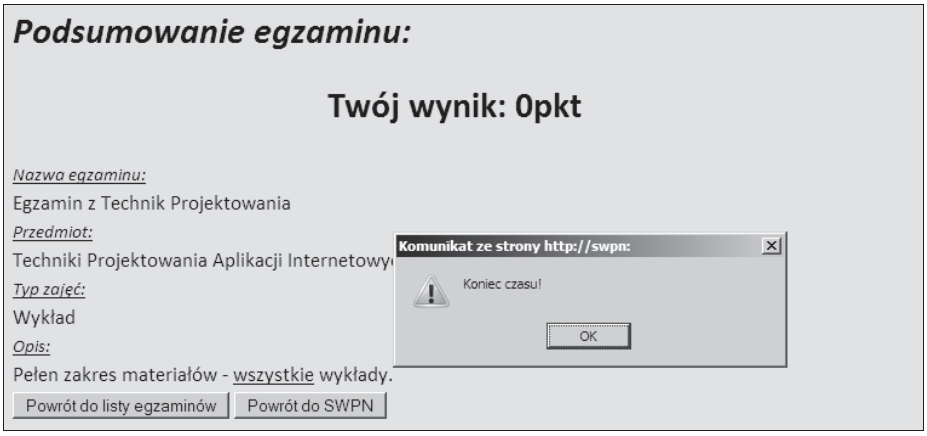
Obsługa procesu modyfikacji (dodawania) pytań, jak również innych danych, opiera swoje działanie na obsłudze zdarzeń generowanych przez te kontrolki [5]. Obsługa realizowana jest poprzez metody udostępniane przez usługę sieciową [7]. Metody te, odwołując się bezpośrednio do bazy danych, realizują modyfikację odpowiednich wpisów.

Wykorzystanie obiektów *UserControl* pozwala na wielokrotne użycie tego samego komponentu w wielu formularzach. Aby skorzystać z komponentu, należy go wskazać (zarejestrować), a następnie dodać do formularza, tak jakbyśmy dodawali zwykłą kontrolkę.

Na rys. 11 przedstawiono interfejs widoczny dla użytkownika egzaminowanego, zwany interfejsem egzaminowania. Interfejs ten składa się z następujących elementów: panelu nawigacyjnego (oznaczenie 1 na rys. 11), panelu informacyjnego (oznaczenie 2 na rys. 11) oraz obszaru przedstawiającego treść pytania wraz z odpowiedziami (oznaczenie 3 na rys. 11).

Rys. 11. Interfejs egzaminowania

Po zakończeniu egzaminu użytkownikowi egzaminowanemu zostaje wyświetlony panel przedstawiający uzyskane przez niego wyniki. Postać panelu, z przykładowymi wynikami, została przedstawiona na rys. 12. Egzamin kończy się, gdy student naciśnie przycisk **Zakończ** lub automatycznie po wyczerpaniu limitu czasu przeznaczonego na egzamin.



Rys. 12. Zobrazowanie przykładowych wyników egzaminu

Podsumowanie

Przedstawiony w artykule System Zdalnego Egzaminowania został wykonany jako moduł rozszerzający dla opracowanego w 2010 roku elektronicznego „Dziennika Nauczyciela". Budowa systemu, jako nieautonomicznego elementu wchodzącego w skład istniejącego narzędzia, była podyktowana kilkoma przesłankami.

Podstawowym czynnikiem przemawiającym za integracją SZE z „Dziennikiem Nauczyciela" była możliwość wykorzystania mechanizmów uwierzytelniania i autoryzacji udostępnionych przez SRU. Użytkownik, używając jednego żetonu uwierzytelnienia, ma możliwość pracowania zarówno w SZE jak i w SWPN. Uprawnienia, które uzyskuje użytkownik, zależne są od typu konta, na które jest zalogowany. Przykładowo użytkownik zalogowany na konto z uprawnieniami *studenta* ma możliwość podglądu swoich dotychczasowych wyników i uruchomienia testu, o ile taki jest mu udostępniony. Użytkownik zalogowany na konto z uprawnieniami *nauczyciela* może tworzyć testy (określając ich strukturę) oraz wystawiać oceny studentowi.

Kolejnym czynnikiem, który wpłynął na wybór takiego rozwiązania był sposób przekazywania oraz przechowywania wyników testów. Wyniki testów są bezpośrednio zapisywane w bazie danych SWPN, gdzie student może je obejrzeć. Z punktu widzenia konta nauczyciela nie ma potrzeby ręcznego wpisywania wyników testów do SWPN. W automatyzacji procesu dodawania ocen do SWPN nieznacznie rozbudowano moduł SZZD, dodając dwie klasy. Pierwsza o nazwie *QuestionAdapter* udostępnia zestaw metod do obsługi (zapisu/odczytu do/z bazy danych) pytań, druga o nazwie *ExaminationAdapter* udostępnia zestaw metod umożliwiających przeprowadzenie testu.

Literatura

[1] PIOTROWICZ M., *System zdalnego egzaminowania*, Praca dyplomowa, Wydział Cybernetyki WAT, 2011.

[2] KWIATEK P., CHUDZIKIEWICZ J., *Projekt scentralizowanego, internetowego systemu rejestracji użytkowników*, „Biuletyn IAiR", 29/2010, str. 19 – 37.

[3] POLKOWSKI S., *System Zdalnego Zarządzania Współdzielonymi Danymi oparty o usługi WWW*, Praca dyplomowa, Wydział Cybernetyki WAT, 2010.

[4] LITWINIUK P., CHUDZIKIEWICZ J., *System Wspomagania Pracy Nauczyciela*, „Biuletyn IAiR", 29/2010, str. 3 – 18.

[5] MAYO J., *C# 3.0 dla .NET 3.5, Księga Eksperta*, Helion, Gliwice, 2010.

[6] DĄBROWSKI W., STASIAK A., WOLSKI M., *Modelowanie systemów informatycznych w języku UML 2.1*, PWN, Warszawa, 2009.

[7] SHEPHERD G., *ASP.NET 3.5, Step by Step*, Microsoft Press, Washington, 2008.

[8] GIBBS M., WAHLIN D., *ASP.NET 2.0 AJAX*, Helion, Gliwice, 2007.

[9] MEYER E., *CSS według Erica Meyera*, Helion, Gliwice, 2005.

Examining remote system

ABSTRACT: A design description including implementation of selected elements of an examining remote system is presented in the paper. Mechanisms of database maintenance using stored procedures are shown. An integration of the elaborated system with the existing system of teacher's diary has been discussed. An explanation to the use of application is given.

KEYWORDS: web application, network service, student's knowledge verification, stored procedures, UML, ASP.NET.

Praca wpłynęła do redakcji: 19.12.2011