Tworzenie produkcyjnego oprogramowania w środowisku akademickim

Janina Mincer-Daszkiewicz*
Instytut Informatyki, Uniwersytet Warszawski

Streszczenie

Głównym zadaniem dydaktyki w zakresie inżynierii oprogramowania jest kształcenie praktyków przygotowanych do brania udziału w rzeczywistych projektach, zaś głównym problemem jest przepaść pomiędzy tą praktyką a projektami prowadzonymi w środowisku akademickim w ramach zajęć dydaktycznych.

Na Wydziale Matematyki, Informatyki i Mechaniki UW studenci uczą się metod i narzędzi inżynierii oprogramowania uczestnicząc w dużym projekcie programistycznym, którego celem jest stworzenie dla polskich uczelni oprogramowania do obsługi spraw studiów. System USOS (Uniwersytecki System Obsługi Studiów) w chwili obecnej działa na dwóch wydziałach dwóch polskich uniwersytetów i wkrótce będzie wdrażany na kilku następnych.

Uczelnia zyskuje dobre oprogramowanie, a studenci praktykę zawodową w zakresie inżynierii oprogramowania.

1. Wprowadzenie

W 1998 roku odbyło się w Poznaniu międzynarodowe sympozjum poświęcone edukacji w zakresie inżynierii oprogramowania ([1]). Lektura artykułów, zarówno tych zaproszonych, jak i zgłoszonych, nie pozostawia żadnych wątpliwości: głównym zadaniem dydaktyki w zakresie inżynierii oprogramowania jest kształcenie praktyków przygotowanych do brania udziału w rzeczywistych projektach, zaś głównym problemem jest przepaść pomiędzy tą praktyką a projektami prowadzonymi w środowisku akademickim w ramach zajęć dydaktycznych. Programy studiów informatycznych dobrze sprawdzają się w zakresie nauczania podstaw. Środowiska akademickie szybko reagują na pojawianie się nowych metodologii, technik i narzędzi. Powszechna jest również świadomość tego, że elementem dydaktyki musi być nie tylko nauka metod i narzędzi inżynierii oprogramowania, ale także wykształcenie umiejętności pracy w zespole, pilnowania terminów, rozmawiania z klientem, prezentacji własnych projektów, poprawiania cudzego kodu, a więc ogólnie uczestniczenia we wszystkich etapach

*Praca częściowo finansowana z grantu Tempus NET UM_JEP-14461-1999.

produkcji oprogramowania: specyfikacji wymagań, projektowaniu, implementacji, testowaniu, wdrażaniu i pielęgnacji.

Potrzeby i oczekiwania przemysłu informatycznego są dobrze określone. Inżynier oprogramowania powinien wiedzieć, że ([6]):

1. trzeba myśleć o systemie jako o całości, a nie tylko o oprogramowaniu; konieczne jest uwzględnienie w projekcie ograniczeń i uwarunkowań sprzętowych i systemowych;

2. wymagania zawsze wiążą się z jakimś kompromisem, nie wszystkich użytkowników systemu można usatysfakcjonować w jednakowym stopniu;

3. nie da się w praktyce całkowicie odseparować etapu formułowania wymagań i projektowania;

4. w przypadku złożonych systemów istotny wpływ na wymagania mogą mieć czynniki społeczne i polityczne;

5. jest wiele możliwych wyborów w zakresie technologii, których trzeba dokonać już na wczesnych etapach budowy systemu, a które mają znaczący wpływ na wymagania i projekt;

6. projekt systemu musi uwzględniać obowiązujące prawo;

7. kwestie wdrożenia trzeba wziąć pod uwagę już na etapie inżynierii wymagań.

Ze świadomości takich oczekiwań biorą się w programach studiów zajęcia, w ramach których studenci w zespołach 3-5 osobowych realizują projekty średniej wielkości. Nie daje to jednak spodziewanych rezultatów. Sztucznie wykreowany problem i warunki takimi pozostają. Studenci są w stanie przyswoić sobie metody i techniki, nie są jednak wystawiani na prawdziwe problemy, nieodłącznie związane z każdym rzeczywistym projektem. Co gorsza, może to wyrabiać w nich przekonanie, że świat realny jest od takich problemów wolny, że każdy projekt programistyczny daje się zrealizować w ciągu semestru, wymagania są z góry znane, nie ma nieoczekiwanych zmian specyfikacji, warunków sprzętowych i programowych, a przekroczenie terminu jest akceptowalne i nie wiąże się z przykrymi konsekwencjami. Studenci odnoszą mylne wrażenie o mocy stosowanych technik (gdyż problem najczęściej dobiera się właśnie pod kątem stosowanych metod), nie rozwijają więc umiejętności właściwej oceny zakresu ich stosowalności. To zaś grozi nierozumieniem potrzeby stosowania technik inżynierii oprogramowania. Istotny jest także aspekt psychologiczny: student nie potrafi wykrzesać z siebie prawdziwego zaangażowania, gdy wie, że zleceniodawca w rzeczywistości nie jest zainteresowany efektami jego pracy.

Przekonanie o złym przygotowaniu absolwentów uczelni do pracy w rzeczywistym środowisku jest powszechne. Przedstawiciele przemysłu informatycznego skarżą się na brak praktyki zawodowej absolwentów studiów informatycznych oraz dużą dysproporcję pomiędzy ich wiedzą teoretyczną a umiejętnościami praktycznymi. Główne zastrzeżenia dotyczą:

- braku umiejętności pracy w zespole, utrzymywania właściwych kontaktów interpersonalnych,
- braku umiejętności pracy w warunkach reżimu czasowego,
- trudności z adaptacją do zmieniających się wymagań,
- braku umiejętności projektowania z myślą o zmianach,
- braku dostatecznej dbałości o jakość produktu,
- braku umiejętności klarownego wyrażania myśli w mowie i piśmie (co przekłada się m.in. na nieumiejętność dokumentowania procesu produkcji programowania na kolejnych jego etapach).

Na niektórych uczelniach podejmuje się próby rozwiązania problemu przez organizowanie projektów we współpracy z przemysłem informatycznym. Projekty takie mają wiele zalet, ale równocześnie niosą ze sobą pewne niebezpieczeństwa. Priorytety firmy są z założenia inne niż priorytety uczelni. Student jest odciągany od uczelni, służbowe wyjazdy i rozmowy z klientami nie są skoordynowane z przebiegiem procesu dydaktycznego, wdrożenie może wymagać zaangażowania uniemożliwiającego studentowi godzenie nauki z realizacją projektu. Cele zewnętrznej firmy mogą ulec zmianie, co grozi przerwaniem projektu. Kontakty z przemysłem mogą być związane z uwarunkowaniami finansowymi. Poza tym w Polsce brak jest jeszcze szerszego zainteresowania wspólnymi projektami ze strony polskich firm, a zachodnie nie są gotowe do takiej współpracy. Pojawiają się one na polskich uczelniach, ale jak na razie w poszukiwaniu pracowników, a nie z wolą współuczestniczenia w procesie dydaktycznym.

Pożądane byłoby rozwiązanie polegające na takiej organizacji projektu, która pozwoli, nie tracąc korzyści wynikających z rzeczywistego charakteru projektu (świadomości, że produkt MUSI powstać), nie wyprowadzać go poza uczelnię, by zminimalizować trudności organizacyjne wynikające z zależności od jednostki pozauczelnianej i utraty pełnej kontroli nad realizacją projektu.

Pomysł polega na organizacji projektów realizowanych przez studentów i polegających na produkowaniu oprogramowania na uczelni i dla uczelni.

Rozwiązanie takie ma wiele zalet. Student obraca się w ramach problematyki świata rzeczywistego, która jest mu w miarę bliska, na miejscu ma klienta, z którym uzgadnia specyfikację wymagań, nie musi opuszczać uczelni, żeby wdrażać swój produkt. Uczelnia ma zaś pełną kontrolę nad przebiegiem procesu produkcji oprogramowania, łącznie z etapem ostatnim: jego pielęgnacją. Studenta można włączać do takich prac już na pierwszych latach, stopniowo zwiększając wymagania, wraz z nabywaniem przez niego nowych umiejętności na kolejnych etapach nauki. Można tworzyć zespoły międzypokoleniowe, zlecając studentom wykonywanie różnych ról w różnych okresach i podzespołach. Jeśli student będzie wciągany do takich prac już na drugim ew. trzecim roku studiów, to okres jego uczestnictwa w projekcie może wynosić od semestru do 3–4 lat.

Co więcej, student może uczestniczyć w projekcie w ramach różnych zajęć dydaktycznych. Zwykle na drugim roku poznaje metody inżynierii oprogramowania, na trzecim bazy danych, programowanie systemowe i sieciowe — wtedy też pisze pracę licencjacką. Na etapie magisterskim uczestniczy w wielu monograficznych zajęciach laboratoryjnych oraz pisze pracę magisterską. Jeśli zatem na młodszych latach pozna narzędzia i metody, wciągnie się w projekt realizując w jego ramach różne drobne prace, a przy okazji poznając jego zasady organizacyjne i standardy, to na etapie magisterskim będzie już przygotowany do samodzielnej realizacji większego modułu.

W dalszej części artykułu przedstawiamy duży wieloletni projekt programistyczny realizowany na Wydziale Matematyki, Informatyki i Mechaniki Uniwersytetu Warszawskiego przez duży zespół studentów różnych lat kierunku informatyka.

2. Uniwersytecki System Obsługi Studiów — USOS

2.1. Projekt Tempus NET

W 1998 roku wszystkie polskie uniwersytety przystąpiły do realizacji projektu Tempus, sponsorowanego przez Unię Europejską w ramach grantu JEP-14461-1999. Jednym z czterech zadań projektu, określonym nazwą NET (od New Educational Tools), było stworzenie oprogramowania do obsługi spraw studiów ([8]). Podstawowym oczekiwaniem grantodawcy było wprowadzenie jednolitego oprogramowania we wszystkich partycypujących w projekcie uczelniach. Zakup gotowej aplikacji do obsługi spraw studiów okazał się jednak niemożliwy z następujących powodów:

1. żaden z oferowanych pakietów komercyjnych nie nadawał się do obsługi spraw studiów na poziomie uczelni,

2. nawet gdyby taki system istniał, to ilość środków przeznaczona na to w grancie nie pozwalałaby na zakup wystarczającej liczby licencji do zaspokojenia choćby podstawowych potrzeb partycypujących uczelni,

3. fundusze nie pozwalały również na zlecenie realizacji takiego oprogramowania firmie komercyjnej.

Z tego powodu podjęto decyzję o wyprodukowaniu oprogramowania własnymi siłami polskich uczelni. Zadania podjął się Wydział Matematyki, Informatyki i Mechaniki Uniwersytetu Warszawskiego. W grudniu 1999 roku rozpoczęto realizację projektu USOS ([9]). Prace są wykonywane przez pracowników, doktorantów i studentów Wydziału. Zdecydowana większość prac projektowych i implementacyjnych jest wykonywana bezpłatnie, w ramach zajęć dydaktycznych, prac licencjackich lub magisterskich.

2.2. Zadania USOS

Od początku założono, że tworzony system ma realizować, w sposób kompleksowy, następujące zadania.

1. Wspomaganie administracyjnej obsługi spraw studenckich: rejestracja na zajęcia, tworzenie grup i rozkładów zajęć; zaliczanie zajęć, przedmiotów, lat studiów; przyznawanie dyplomów; pomoc socjalna; opłaty.
2. Wspomaganie dydaktyki: wybór przedmiotów; uaktualnianie programów studiów; ułatwianie studiów interdyscyplinarnych i mobilności studentów; obsługa punktowa zgodna z ECTS.
3. Informowanie: oferta dydaktyczna uczelni, przebieg procesu kształcenia, rozwiązania organizacyjne.
4. Usprawnienie działania i obniżenie kosztów: automatyzacja masowej obsługi, narzędzia do analizy kosztów organizacji dydaktyki.

2.3. Podstawowe założenia projektowe USOS

Oto główne założenia projektowe systemu.

1. Zakres: system do obsługi spraw studiów (nie obejmuje kadr, finansów i innych spraw nie związanych bezpośrednio z dydaktyką).
2. Zasięg: ogólnouniwersytecki (istnienie jednej bądź wielu instalacji bazy w ramach uczelni, z możliwością automatycznego lub ręcznego przesyłania danych), możliwość utrzymywania wersji “pełnej” i wersji “oszczędnej” (z okrojoną funkcjonalnością).
3. Elastyczność

   System musi być na tyle elastyczny, by mógł być stosowany przez wszystkie polskie uczelnie partycypujące w projekcie NET. Ze względu na duże różnice w sposobie organziacji dydaktyki na różnych uczelniach, system musi pozwalać na konfigurowanie podstawowych elementów struktury organizacyjnej uczelni i przebiegu procesu dydaktycznego. Oto przykładowe elementy konfiguracji:

   (a) definiowanie struktury uczelni — wydziały, instytuty, zakłady, katedry;
   (b) definiowanie jednostek i okresów zaliczeniowych — lata, semestry, trymestry — i przypisywanie im zajęć;
   (c) definiowanie kierunków i toków studiów niezależnie od struktury uczelni;
   (d) tworzenie w ramach jednego kierunku ścieżek odpowiadających różnym specjalnościom; definiowanie warunków minimalnych dla wejścia na daną ścieżkę;
   (e) definiowanie skali ocen i wagi każdej oceny w średniej; sposobów naliczania średnich, typu i liczby protokołów egzaminacyjnych;
   (f) dowolne konfigurowanie systemu zaliczeń poszczególnych etapów studiów, np. na podstawie: liczby punktów (różnych rodzajów), uzyskania określonego poziomu w ramach grupy przedmiotów (np. 4 poziom angielskiego), spełnienia dowolnego warunku logicznego (np. oddanie książki do biblioteki);

(g) definiowanie list przedmiotów równoważnych w ramach programów;
(h) definiowanie certyfikatów związanych z etapami studiów bądź przedmiotami;
(i) przygotowywanie zestawień obejmujących zasięgiem całą uczelnię (każdy student pojawia się w systemie tylko raz);
(j) rozliczanie obciążeń dydaktycznych na dowolnym poziomie — jednostki, wydziału, uczelni.

4. Otwartość: ze względu na częste zmiany w programach studiów system musi pozwalać na łatwe dodawanie nowych funkcji i modyfikowanie starych.

5. Dostęp: wygodny, przyjazny, graficzny; w wybranym zakresie i dla wybranych grup użytkowników dostęp masowy przez WWW.

6. Bezpieczeństwo: wielopoziomowy system uprawnień.

Dodatkowym wymaganiem, zgłoszonym przez wiele uczelni, była zgodność ze standardami ECTS.

2.4. Architektura systemu

USOS ma z jednej strony wspomagać administracyjną obsługę studiów, a z drugiej pełnić rolę informacyjną w stosunku do wszystkich osób zaangażowanych w proces dydaktyczny. Takich osób jest jednak bardzo dużo — średniej wielkości Wydział ma dwustu-trzystu pracowników i ponad tysiąc studentów, a średniej wielkości uczelnia ma tysiące pracowników i kilkanaście tysięcy studentów. Ważnym aspektem działania systemu jest sprawne obsługiwanie wszystkich tych osób, z zachowaniem poufności danych, bezpieczeństwa i efektywności.

Dla rozwiązania tych problemów podjęto decyzję, że USOS w sensie logicznym będzie się składał z dwóch części: działającej w uczelnianej sieci komputerowej (moduł główny) i dostępnej przez Internet (moduł internetowy).

Część działająca w sieci uczelnianej będzie obejmowała centralną bazę danych oraz stanowiska klienckie, poprzez które będzie realizowany ściśle limitowany dostęp do tej bazy z wyznaczonych stanowisk na terenie uczelni: Biura Spraw Studenckich, Biura Spraw Pracowniczych (uczelnianego i wydziałowych), Sekcji Studenckich dziekanatów na wydziałach, jednostek odpowiedzialnych za ofertę dydaktyczną wydziału (np. sekretariaty instytutów). Funkcje systemu udostępniane przez moduł główny są przeznaczone dla wąskiego grona administracyjnych pracowników uczelni, zgodnie z przypisanymi im rolami.

Część działająca w sieci Internet, tzw. USOSweb ([10]), ma bezpośredni dostęp jedynie do pomocniczych internetowych baz danych. Daje ona nieograniczony (przestrzennie) personalizowany dostęp do bazy danych modułu internetowego z dowolnie wielu, dowolnie rozproszonych stanowisk, dla dużych grup użytkowników — studentów, nauczycieli akademickich, czy świata zewnętrznego (w tym kandydatów na studia).

USOSweb udostępnia następujące usługi:

- informacja o ofercie dydaktycznej uczelni (dostęp bez ograniczeń); obejmuje ona:

  1. katalog przedmiotów: lista przedmiotów z ich atrybutami, system wyszukiwania przedmiotów wg atrybutów, przyporządkowanie przedmiotów jednostkom organizacyjnym, lista programów studiów;
  2. oferowane certyfikaty, zbiór przedmiotów, kryteria zaliczeń, podział na okresy dydaktyczne;
  3. plan zajęć, dostępność prowadzących (konsultacje);

- rejestracja na przedmioty i zajęcia połączona z dopasowaniem programu rejestracyjnego studenta do wymagań etapu studiów (dostęp dla studentów),
- elektroniczny indeks (dostęp dla studentów),
- elektroniczne protokoły egzaminów i zaliczeń (dostęp dla pracowników).

USOSweb daje możliwość pracy bez stałego połączenia z bazą centralną (komunikacja asynchroniczna); wymiana informacji może się odbywać automatycznie w ustalonych odstępach czasu. Baza danych USOSweba zawiera dane importowane z bazy centralnej (katalogi programów i przedmiotów, plan zajęć, listy zaliczeń studentów itp.) oraz dane wprowadzane przez Internet i eksportowane do bazy centralnej (listy rejestracyjne, elektroniczne protokoły itp.)

Takie rozwiązanie ma wiele zalet: bezpieczeństwo, efektywność dostępu, uniezależnienie od licencji bazodanowych. USOSweb jest zabezpieczony przed niepowołanym dostępem, np. hasła i identyfikatory użytkowników są przesyłane siecią w postaci zakodowanej.

3. Wybrane aspekty realizacji projektu USOS

3.1. Metodyka i stosowane narzędzia

Część środków Unii Europejskiej przeznaczono na kupno licencji na system bazy danych dla partycypujących w projekcie uczelni oraz na narzędzia projektowe i implementacyjne. Wybrane narzędzia musiały zapewniać wsparcie dla nowoczesnych technik inżynierii oprogramowania oraz być stosowane w dydaktyce.

Jako platformę bazodanową dla projektu wybrano Oracle 8.0.5. Ta decyzja zdeterminowała zarówno dobór narzędzi projektowych (Oracle Designer/2000) i implementacyjnych (Oracle Developer, Oracle Forms i Oracle Reports), jak i metodyki prowadzenia projektu (opracowana w firmie Oracle Custom Development Method — CDM). Dostępność takich narzędzi oraz stowarzyszonej z nimi metodologii była jednym z głównych powodów wyboru platformy Oracle.

Użycie narzędzia Oracle Designer znacznie ułatwiło realizację szeregu określonych przez CDM zadań, takich jak modelowanie procesów i funkcji, stworzenie logicznego i fizycznego modelu danych. Ze względu na przechowywanie danych o projekcie w repozytorium umieszczonym w bazie Oracle, Designer/2000 umożliwiał równoczesną pracę nad projektem wielu analitykom, projektantom i programistom. Ponadto można było generować formularze Oracle Forms na podstawie zaprojektowanych w nim modułów aplikacji. Automatyczna generacja formularzy znacznie przyśpiesza proces implementacji funkcjonalności systemu. Zmiany w modelu danych i modelu funkcjonalności nie wymagają gruntownych modyfikacji kodu formularzy — proces generacji można wówczas powtórzyć. Uzyskane w ten sposób formularze były następnie adaptowane do obowiązujących w projekcie standardów.

Na potrzeby modułu internetowego wybrano oprogramowanie niekomercyjne (MySQL, PHP, Apache), m.in. w celu uniknięcia płacenia wysokich opłat licencyjnych wynikających z masowego dostępu do modułu.

Z innych metodologii stosowanych na etapie analizy wymagań oraz projektowania można wymienić Przypadki Użycia oraz UML. W fazie implementacji i wdrażania dominowało szybkie prototypowanie. Istotną rolę w zarządzaniu kodem źródłowym pełnił system kontroli wersji CVS.

3.2. Analiza wymagań

Ponieważ celem projektu było opracowanie systemu na potrzeby różnych uczelni wyższych, dlatego na etapie specyfikacji wymagań zdecydowano się na realizację narzędzia wspomagającego zbieranie postulatów odnośnie funkcjonalności systemu. Zbudowano tzw. *Postulator*, prostą aplikację internetową umożliwiającą wprowadzanie postulatów za pomocą przeglądarki WWW, modyfikowanie wcześniej wprowadzonych postulatów (takie uprawnienia miał jedynie autor postulatu) oraz przeglądanie wszystkich wcześniej wprowadzonych. Pomocne na tym etapie okazało się również zgromadzenie informatorów i regulaminów studiów różnych jednostek dydaktycznych.

3.3. Organizacja pracy

Przy realizacji tak dużego projektu, łączącego cele dydaktyczne i produkcyjne, kluczową rolę spełnia właściwa organizacja pracy.

Rolę kierownika projektu pełni prodziekan Wydziału ds informatyzacji i organizacji. Na stałe w projekcie uczestniczy dwóch projektantów, będących pracownikami naukowo-dydaktycznymi Wydziału. Role liderów grup oraz programistów pełnią studenci. Dzięki temu, że repozytorium projektu jest przechowywane w narzędziach CASE, wprowadzanie do projektu nowych osób jest w miarę proste. Każdy nowy członek zespołu na początek dostaje do wykonania proste prace programistyczne. Podczas okresu terminowania poznaje model logiczny i fizyczny bazy, szlifuje swoją znajomość narzędzi projektowych i programistycznych, poznaje obowiązujące w projekcie standardy oraz organizację pracy. Po okresie wstępnym dostaje pierwsze zadanie, bądź samodzielne, bądź w grupie 2-3 osobowej. Pierwsze większe samodzielne zadanie to zazwyczaj praca licencjacka bądź zalążek pracy magisterskiej. Wraz z nabywaniem doświadczenia student awansuje w projekcie, uzyskuje coraz więcej samodzielności i dostaje coraz większe uprawnienia.

Starsi stażem studenci pełnią w projekccat caie bardziej odpowiedzialne role. Jest administrator projektu odpowiedzialny za nadzór nad wersją produkcyjną oprogramowania oraz przygotowywanie kolejnych wersji dystrybucyjnych. Jest specjalista od raportów, który dba o jednolitą formę wszystkich raportów generowanych z systemu. Inna osoba nadzoruje biblioteki, zestawy preferencji dla generatora formularzy, redagowanie ustaleń standaryzacyjnych. Jeszcze inni pełnią role liderów grup programistycznych, kierujących pracą niewielkich podzespołów. Role pełnione przez studenta zmieniają się w trakcie jego uczestniczenia w projekcie: od wykonawczych, poprzez współprojektowanie, do współzarządzania projektem.

W projekcie uczestniczą także osoby spoza grupy nauczycieli akademickich i studentów. Są to m.in. administrator bazy danych Oracle oraz niezależny tester, który jest równocześnie autorem podręcznika użytkownika systemu.

Tryb pracy nad poszczególnymi elementami systemu jest następujący. Student dostaje do wykonania nowe zadanie. Najpierw pracuje nad projektem. Spotyka się co kilka dni z jednym z projektantów (każdy projektant ma pod swoją opieką równocześnie kilka takich zadań), omawiając z nim projekt. Na tym etapie przygotowuje przypadki użycia i przykłady interfejsu użytkownika. Gdy projekt jest już gotowy, prace implementacyjne są rozbijane na kilka etapów, dla każdego etapu ustala się termin wykonania. Zwykle etapy są krótkie i obejmują 1–2 tygodni. Jeśli zadanie wymaga zmian w istniejącej strukturze bazy, to student jest obowiązany przygotować oficjalny dokument zgłoszenia zmian, zawierający: listę zmian, uzasadnienie, informację o wpływie zmian na inne części systemu (ang. *impact analysis*) oraz skrypt zmieniający bazę. Wyniki etapu są odbierane przez nadzorującego zadanie projektanta. Teraz do pracy przystępuje tester, który bada poprawność i zgodność ze standardem. Gotowa część zostaje przekazana klientowi (mamy więc do czynienia z szybkim prototypowaniem). Usterki są zgłaszane autorom, którzy usuwają błędy na bieżąco. Czasami konieczna jest pielęgnacja cudzego kodu (gdyż studenci po obronie pracy magisterskiej odchodzą z projektu).

Studenci pracując nad systemem, którego poszczególne elementy są wdrażane na bieżąco, mają świadomość jak ważne jest dotrzymywanie terminów wykonania poszczególnych prac.

Podjęto wiele działań mających na celu poprawę jakości i organizacji pracy. Dla studentów realizujących kluczowe zadania przygotowano na Wydziale osobne stanowiska pracy (pozostali studenci korzystają z ogólnodostępnych laboratoriów). Studenci mogą na bieżąco prosić o pomoc merytoryczną, zarówno projektantów, jak i starszych kolegów. Uruchomiono listy dyskusyjne i aliasy pocztowe ułatwiające wymianę pomysłów, uzgodnień oraz uzyskiwanie pomocy technicznej. Stworzono specjalny serwis internetowy udostępniający dokumentację projektu, opisy standardów, harmonogram prac, zobowiązania poszczególnych osób. W szczególności każdy student ma w serwisie swoją stronę z opisem wykonywanych prac, terminów, wykrytych błędów (taki prosty system archiwizowania błędów okazał się wystarczający). Systematycznie odbywają się spotkania wszystkich członków zespołu z kierownikiem projektu.

Studenci, oprócz prezentowania swoich prac na zajęciach, np. w ramach referatów na seminarium magisterskim, biorą też udział w ogólnopolskich seminariach projektu

Tempus, na których prezentują je władzom Wydziału, uczelni, a także zaproszonym gościom z innych uczelni. Takie seminaria działają bardzo mobilizująco, a ponadto powodują, że student utożsamia się z projektem, czuje odpowiedzialność za jakość swojej pracy i niewątpliwie także satysfakcję.

3.4. Przystępowanie studentów do projektu

Udział studentów w projekcie jest dobrowolny. Jeśli nawet jakiś fragment systemu powstaje w ramach zajęć obowiązkowych, to dołączenie tego fragmentu do wersji produkcyjnej jest już realizowane tylko przez chętnych. W chwili obecnej w różnych pracach na rzecz systemu uczestniczy około 30 studentów. Motywacje studentów przystępujących do projektu są różne. Niewątpliwie najważniejsza jest możliwość uzyskania dyplomu licencjackiego lub magisterskiego, czy zaliczenie konkretnych zajęć. Student ceni sobie to, że od razu uzyskuje indywidualną opiekę, ma dobrze określone zadanie i ustalony harmonogram prac. Wbrew obiegowej opinii działanie pod presją czasową jest dla niego korzystne, gdyż dopinguje go do wywiązywania się z prac w terminie (jest to istotny argument w sytuacji, gdy jeszcze do niedawna prace magisterskie broniło w terminie przeciętnie 20% absolwentów kierunku informatyka).

Studenci myślą bardzo praktycznie o przyszłej pracy. Ma więc dla nich znaczenie, że mogą dobrze poznać narzędzia projektowe i implementacyjne, które są cenione przez przyszłych pracodawców. Wiedzą, że udział w tak dużym projekcie będzie się liczył w ich życiorysie zawodowym.

Cenione są także ułatwienia w dostępie do sprzętu na Wydziale. Nie bez znaczenia jest satysfakcja zawodowa i partnerskie stosunki z nauczycielami akademickimi. Zdarza się, że do projektu przystępują studenci nie oczekujący gratyfikacji w postaci zaliczenia zajęć czy uzyskania certyfikatu.

Udział w projekcie takiego typu można również powiązać z jakimiś korzyściami materialnymi studentów. Mogłaby to być symboliczna pensja (w projekcie USOS taką dostają studenci wykonujący bezpośrednie prace administracyjne i wdrożeniowe), czy zwolnienie z opłat na rzecz uczelni.

4. USOS dziś

W chwili obecnej istnieje już pilotażowa wersja USOS, z mocno jeszcze okrojoną, ale już dostateczną funkcjonalnością, która jest wdrażana na Wydziale MIM UW, na Wydziale Matematyki Uniwersytetu Łódzkiego oraz na Wydziale Filozofii Uniwersytetu im. Kardynała Stefana Wyszyńskiego. Na Uniwersytecie Warszawskim zapadła już decyzja o tym, że USOS zostanie wprowadzony jako system ogólnouniwersytecki, do jego wdrażania na innych wydziałach przygotowuje się powołany przez rektora zespół. Inne uczelnie na razie przyglądają się rozwojowi systemu, coraz więcej ich deklaruje jednak wolę wdrożenia. Wkrótce ma powstać konsorcjum polskich uczelni zainteresowanych rozwojem i wdrażaniem systemu, na wzór analogicznego konsorcjum uczelni szwedzkich o nazwie LADOK ([7]).

W ramach projektu USOS powstało już kilka prac magisterskich, na ukończeniu jest kilka następnych (oto przykładowe prace: [3, 5, 4, 2]).

5. Podsumowanie

Zadaniem dobrego kursu z inżynierii oprogramowania jest stworzenie właściwego środowiska i wypracowanie metody nauczania, która pozwala na działanie w warunkach jak najbardziej zbliżonych do rzeczywistych.

Na Wydziale Matematyki, Informatyki i Mechaniki UW studenci uczą się metod i narzędzi inżynierii oprogramowania uczestnicząc w dużym projekcie programistycznym, którego celem jest przygotowanie dla polskich uczelni oprogramowania do obsługi spraw studiów.

Praca studentów na rzecz uczelni nie jest czymś nowy. To czym projekt USOS różni się od innych, to skala oraz wielopoziomowe sprzężenie prac projektowych, implementacyjnych i wdrożeniowych z procesem dydaktycznym.

Po prawie dwóch latach funkcjonowania projektu można dokonać pewnego podsumowania. Realizacja zadania tej skali przy współudziale studentów i sprzężenie tego z dydaktyką ma wiele dobrych stron:

1. studenci uczestniczą we wszystkich etapach rzeczywistego projektu programistycznego, łącznie z wdrożeniem i pielęgnacją (nierzadko cudzego) kodu;

2. studenci uczą się pracy w zespole, także rozproszonym (część prac była wykonana na Wydziale Matematyki Uniwersytetu Łódzkiego), współpracy z kolegami ze studiów dziennych bądź wieczorowych, etapu licencjackiego lub magisterskiego — dzięki temu po skończeniu studiów są dobrze przygotowani do pracy w dowolnych zespołach programistycznych;

3. studenci uczą się używania profesjonalnych narzędzi, spełniania oczekiwań odnośnie wydajności, podporządkowywania się standardom, odpowiedzialności za jakość własnej pracy, za poufność i integralność danych;

4. studenci czują się związani z projektem, nawet w nim bezpośrednio nie uczestnicząc; wystarczy im świadomość, że oprogramowanie powstaje na uczelni, rękoma ich kolegów i że wszyscy mogą współuczestniczyć w specyfikowaniu wymagań;

5. uczelnia zyskuje dobrej jakości oprogramowanie.

Wśród zagrożeń takiego projektu można wymienić następujące:

1. wymaga on zdecydowanie większego zaangażowania ze strony nauczycieli akademickich, znacznie wykraczającego czasowo poza wymagane pensum dydaktyczne;

2. konieczność stawiania celów dydaktycznych na pierwszym miejscu może być trudna do pogodzenia z potrzebą szybkiego wdrożenia, np. ze względu na konieczność zamrożenia prac w okresie sesji egzaminacyjnych, opóźnienia prac projektowych związanego z próbą zastosowania, w celach dydaktycznych, różnych metodologii projektowych;

3. mogą być trudności praktyczne ze skierowaniem takiej oferty dydaktycznej do wszystkich (z drugiej strony tak samo jest z zajęciami monograficznymi — na te najbardziej popularne trafiają nieliczni);

4. dokumentacja projektowa i techniczna systemu w postaci prac licencjackich i magisterskich częściowo tylko spełnia potrzeby produkcyjne.

Bibliografia

[1] P. Klint i J.R. Nawrocki, editor. *Proc. Software Engineering Education Symposium SEES'98.* Scientific Publishers OWN, Poznań, 1998.

[2] J. Nowisz i M. Olszewik. Usosweb. Moduł internetowy Uniwersyteckiego Systemu Obsługi Studiów. Praca magisterska (w przygotowaniu), Instytut Informatyki Uniwersytetu Warszawskiego.

[3] J. Korkuć i R. Rudzki. Uniwersytecki System Obsługi Studiów. Praca magisterska, Instytut Informatyki Uniwersytetu Warszawskiego, 2000.

[4] T. Juśkiewicz i T. Krajewski. Uniwersytecki System Obsługi Studiów. Rejestracja i zaliczenia. Praca magisterska (w przygotowaniu), Instytut Informatyki Uniwersytetu Warszawskiego.

[5] A. Sawicki. System Obsługi Spraw Pracowniczych. Praca magisterska, Instytut Informatyki Uniwersytetu Warszawskiego, 2000.

[6] I. Sommerville. Learning from the Impossible: the Case for Wicked Problems in Teaching Systems Engineering. In P. Klint i J.R. Nawrocki, editor, *Proc. Software Engineering Education Symposium SEES'98*, pages 275–283, Poznań, 1998. Scientific Publishers OWN.

[7] Stron domowa konsorcjum LADOK, Umea, Szwecja. `http://www.ladok.umu.se`.

[8] Stron domowa projektu Tempus NET — New Educational Tools. `http://www.mimuw.edu.pl/NET`.

[9] Stron domowa projektu USOS — Uniwersytecki System Obsługi Studiów. `http://www.mimuw.edu.pl/USOS`.

[10] Stron domowa serwisu USOSweb na Wydziale Matematyki, Informatyki i Mechaniki Uniwersytetu Warszawskiego. `http://usosweb.mimuw.edu.pl`.