SPOŁECZNA AKADEMIA NAUK W ŁODZI

Sylabusy specjalizacji „TECHNOLOGIE PROGRAMOWANIA”

„informatyka”
studia pierwszego stopnia (inżynierskie)
o profilu ogólnoakademickim

SPECJALNOŚĆ: TECHNOLOGIE PROGRAMOWANIA

PROGRAMOWANIE WSPÓŁBIEŻNE

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) PROGRAMOWANIE WSPÓŁBIEŻNE			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy obowiązkowy	
Rok: 3	**Semestr: 5**	**ECTS ogółem:4**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 2**	**studia niestacjonarne: 2**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Matematyka dyskretna** **Algorytmy i złożoność** **Języki i paradygmaty programowania**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: egzamin** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Konrad Grzanek**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	15	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	20
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	2
RAZEM:	**47**	**RAZEM:**	**32**

Praca własna studenta (PWS):	53	Praca własna studenta (PWS):	68
RAZEM z PWS:	**100**	**RAZEM z PWS:**	**100**

Sumaryczne obciążenie pracą studenta wg form aktywności:

Forma aktywności:	**Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:**	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**47**	**32**
2. Przygotowanie się do zajęć	23	28
3. Przygotowanie esejów		
4. Wykonanie projektów	20	20
5. Zapoznanie z literaturą podstawową	10	20
6. Pisemna praca zaliczeniowa		
7. Inne:		
SUMA:	**100**	**100**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Istota programowania współbieżnego, procesy i wątki w przestrzeni użytkownika a zadania w przestrzeni jądra systemu operacyjnego, wywłaszczanie, dzielenie czasu i znaczenie algorytmu szeregującego.
2. Tworzenie procesów oraz wątków w systemach operacyjnych, realizacja w środowisku UNIX, realizacja na platformie Java.
3. Wątki w języku Java, tworzenie, cykl życia, aspekty składniowe, użycie interfejsu Runnable.
4. Wprowadzenie do aspektów związanych z komunikacją pomiędzy wątkami w języku Java.
5. Pojęcie semafora, historia, implementacje na różnych platformach systemowych i w różnych językach programowania.
6. Sekcja krytyczna, metody realizacji – semafory, monitory, realizacja semaforów i monitorów w języku Java.
7. Problem czytelników i pisarzy, problem pięciu filozofów, realizacja z wykorzystaniem semaforów – omówienie.
8. Aspekty implementacji dynamicznych struktur danych bezpiecznych z punktu widzenia wielowątkowości w języku Java.
9. Najważniejsze błędy związane z realizacją sekcji krytycznych, zakleszczenie, zagłodzenie, metody zapobiegania powstawaniu tych zjawisk w języku Java. Bezpieczne tworzenie obiektów.
10. Java Memory Model jako formalizacja dotychczasowych rozważań o wielowątkowości w Javie.
11. Zadania w języku Ada 2005. Tworzenie, mechanizm spotkań (rendez-vous).
12. Software transactional memory w języku Clojure.
13. Komunikacja międzyprocesowa w systemie UNIX.
14. Przedstawienie standardowych mechanizmów bibliotecznych (Concurrency Classes) związanych ze współbieżnością w języku Java, pule wątków, obiekty atomowe, timery.
15. Podsumowanie wykładu.

LABORATORIA:

Zaliczenie ćwiczeń laboratoryjnych polega na ocenie zestawu zadań nawiązujących tematycznie do treści wykładów. Przykładowe tematy do opracowania:

1. Implementacja rozwiązania problemu pięciu filozofów w języku Java oraz Ada, implementacja rozwiązania problemu czytelników i pisarzy w języku Java oraz Ada.
2. Implementacja bufora blokującego (BlockingBuffer) w języku Java.
3. Implementacja mechanizmu rozgłoszeniowego dla komunikatów w języku Java.
4. Implementacja wybranego algorytmu wykorzystującego mechanizmy wielowątkowości oraz transakcyjną pamięć współdzieloną w języku Clojure.
5. Implementacja systemu klient-serwer z wykorzystaniem wybranego mechanizmu komunikacji między procesami w systemie UNIX (Linux).

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:

Wiedza:

Kod wg KRK:		**Kod KEK/%: udział przedmiotu w osiągnięciu efektu:**	**Metoda (forma) weryfikacji***
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Test wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego

Umiejętności:

Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	obserwacja i ocena wykonania zadania praktycznego

Kompetencje społeczne:

Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań

* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?

Cele kształcenia (przedmiotowe efekty kształcenia):

Celem przedmiotu jest zapoznanie studenta z najważniejszymi zagadnieniami dotyczącymi wielozadaniowości, w szczególności wielowątkowości i komunikacji międzyprocesowej w wielozadaniowych systemach operacyjnych oraz na platformie Java.

Po ukończeniu kursu student:

- ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie wielozadaniowych systemów operacyjnych, mechanizmów wielozadaniowości i współbieżności, wieloprocesowości i związanych z tym algorytmów
- zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu zadań informatycznych z zakresu programowania współbieżnego
- ma umiejętność formułowania algorytmów rozwiązujących problemy związane z realizacją jednoczesnego dostępu do zasobów współdzielonych przez wątki i procesy

W szczególności student powinien:

- rozumieć zręby zasad działania wielozadaniowych systemów operacyjnych
- znać pojęcia wątku, zadania, procesu
- zdawać sobie sprawę ze znaczenia synchronizacji dostępu do zasobów współdzielonych przez zadania, w szczególności – wątki
- potrafić zbudować i uruchomić wielowątkową aplikację w języku Java, zarządzać czasem życia wątków w Javie
- potrafić prawidłowo zrealizować sekcje krytyczne w języku Java
- być w stanie zaimplementować rozwiązania najważniejszych problemów współbieżności w języku Java (oraz opcjonalnie – Ada)
- znać Java Memory Model, rozumieć jego podstawowe założenia, szczególnie następstwo zdarzeń wynikające z zastosowania słowa synchronized
- znać najważniejsze mechanizmy biblioteczne wspierające tworzenie programów wielowątkowych na platformie Java SE
- potrafić wykorzystywać mechanizmy Software Transactional Memory w języku Clojure
- potrafić budować mechanizmy spotkań w języku Ada
- potrafić zrealizować podstawowe programy wieloprocesowe w systemie UNIX i implementować mechanizmy komunikacji pomiędzy procesami tamże

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— M. Ben-Ari, Podstawy programowania współbieżnego i rozproszonego, WNT, Warszawa 2009, ISBN 978-83-204-3411-5
— W. Richard Stevens, Programowanie w środowisku systemu UNIX, WNT 2002, ISBN 83-204-2669-3
— Joshua Bloch, Brian Goetz, Doug Lea, Tim Peierls, Joseph Bowbeer, and David Holmes, Java Concurrency in Practice, ISBN 0-321-34960-1, 2006

Literatura uzupełniająca przedmiotu:

— Doug Lea, Concurrent Programming in Java: Design Principles and Patterns, first edition: 1997; second edition: ISBN 0-201-31009-0, 1999

Inne materiały dydaktyczne:

PROGRAMOWANIE FUNKCYJNE

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) PROGRAMOWANIE FUNKCYJNE			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy obowiązkowy	
Rok: 3	**Semestr: 5**	**ECTS ogółem:4**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 2**	**studia niestacjonarne: 1**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Analiza matematyczna i algebra liniowa. Matematyka dyskretna. Algorytmy i złożoność. Języki i paradygmaty programowania**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: zaliczenie** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Konrad Grzanek**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	15	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	10
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	2
RAZEM:	**47**	**RAZEM:**	**22**
Praca własna studenta (PWS):	53	Praca własna studenta (PWS):	78
RAZEM z PWS:	**100**	**RAZEM z PWS:**	**100**
Sumaryczne obciążenie pracą studenta wg form aktywności:			

Forma aktywności:	Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**47**	**22**
2. Przygotowanie się do zajęć	23	38
3. Przygotowanie esejów		
4. Wykonanie projektów	20	20
5. Zapoznanie z literaturą podstawową	10	20
6. Pisemna praca zaliczeniowa		
7. Inne:		
SUMA:	**100**	**100**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY

1. Computer science, procesy, wiedza deklaratywna i imperatywna, operator punktu stałego – omówienie teoretyczne.
2. Sposoby radzenia sobie ze złożonością, abstrakcja „czarnej skrzynki", konwencjonalne interfejsy.
3. Wprowadzenie do programowania w języku Clojure; obiekty atomowe (primitives) kombinacja (s-wyrażenia), abstrakcja (forma specjalna def/defn).
4. Formy specjalne warunkowe if/cond, predykaty i wartości prawda/fałsz w Clojure, formy and/or, short-circuit evaluation.
5. Wzór Herona – implementacja.
6. Model podstawieniowy ewaluacji wyrażeń.
7. Rodzaje wyrażeń w Lispie (Clojure, Scheme).
8. Reguły ewaluacji wyrażeń.
9. Rekurencja w sensie definicji, przykłady procedur zdefiniowanych z jej wykorzystaniem.
10. Procesy iteracyjne, zmienne stanu, zależności przestrzenne i czasowe.
11. Procesy rekurencyjne, stos wywołań, zależności przestrzenne i czasowe.
12. Rekurencja krańcowa i jej optymalizacja, forma recur w Clojure.
13. Ciąg Fibonacciego.
14. Procedury wyższego rzędu w Clojure.
15. Operator punktu stałego – implementacja w Lispie.
16. Wzór Herona – implementacja z wykorzystaniem operatora punktu stałego.
17. Technika pomocnicza: tłumienie przez uśrednianie (ang. average-damping).
18. Różniczkowanie numeryczne – naiwna implementacja z wykorzystaniem procedur wyższego rzędu i
19. formy specjalnej fn.
20. Metoda Newtona – omówienie i implementacja.
21. Dane złożone, realizacja ułamków (ang. ratio) w Clojure.
22. Forma specjalna let.
23. Własność domknięcia (ang. closure).
24. Cytowanie, forma specjalna quote w Lispach.
25. Różniczkowanie symboliczne wyrażeń algebraicznych – omówienie i implementacja.

LABORATORIA

Zaliczenie na podstawie ćwiczeń laboratoryjnych.

Ćwiczenia zadawane są przez prowadzącego na kolejnych zajęciach w semestrze. Powinny odpowiadać merytorycznie treściom wykładowym.

Propozycje zadań:

Zadanie 1.

1. Zaimplementować wzór na pierwiastek sześcienny podobnie do sposobu wykorzystanego we wzorze Herona.

Zadanie 2.

Wyznaczyć dokładny wzór opisujący ilość kroków niezbędnych do obliczenia n-tego wyrazu ciągu Fibonacciego przy założeniu realizacji z rekurencją drzewiastą.

Zaproponować procedurę rekurencyjną Fib, która generuje proces iteracyjny

Zastosować formę (recur ...) i policzyć Fib(10000).

Zadanie 3.

Zrealizuj pierwiastek sześcienny z wykorzystaniem tłumienia przez uśrednianie oraz operatora punktu stałego.

Zrealizuj pierwiastek sześcienny z wykorzystaniem metody Newtona.

Niech f i g będą dwoma funkcjami jednoargumentowymi. Złożenie funkcji f i g jest określone jako funkcja x → g(f(x)). Zaimplementuj procedurę realizującą złożenie funkcji.

Jeśli f jest funkcją jednoargumentową określoną na liczbach a n jest dowolną liczbą naturalną, to n-krotnym złożeniem funkcji f nazywamy funkcję, której wartością jest wynik n-krotnego zastosowania funkcji f: x → f(f(... (f(x)) ...))

Napisz procedurę realizującą n-krotne złożenie funkcji f wykorzystując rozwiązanie z punktu c.

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:

Wiedza:

Kod wg KRK:		**Kod KEK/%: udział przedmiotu w osiągnięciu efektu:**	**Metoda (forma) weryfikacji***
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Test wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych,inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego

Umiejętności:

Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09	ma umiejętność formułowania algorytmów i ich	K_U08/5%	obserwacja i ocena

T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je,odszukać w nich słabości i błędy oraz opracować plan testów		wykonania zadania praktycznego
Kompetencje społeczne:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań

* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?

Cele kształcenia (przedmiotowe efekty kształcenia):

Celem uczestnictwa w zajęciach przedmiotowych jest nabycie umiejętności programowania w funkcyjnym języku ***Clojure*** (nowoczesny dialekt języka ***Lisp***) oraz pozyskanie kompetencji w zakresie dekompozycji problemów o charakterze matematycznym do postaci umożliwiającej osiągnięcie ich rozwiązania z zachowaniem stylu funkcyjnego. Ze względu na utrzymanie spójności merytorycznej przedstawione cele dotyczą zarówno wykładów jak i zajęć laboratoryjnych.

Po ukończeniu kursu student:

- ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie programowania funkcyjnego, w szczególności - programowania symbolicznego w języku programowania Clojure
- zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych związanych ze stosowaniem funkcyjnego stylu programowania
- ma umiejętność formułowania algorytmów i ich implementacji stosując styl funkcyjny, w tym: rekurencję, rekurencję krańcową, wyrażenia warunkowe oraz domknięcia, zna najważniejsze narzędzia programistyczne niezbędne do programowania w języku Clojure

W szczególności student powinien:

- Posiadać umiejętność tworzenia prostych i złożonych procedur w języku Clojure, w tym – procedur wyższego rzędu oraz procedur będących wartościami lambda-wyrażeń.
- Posiadać umiejętność posługiwania się trwałymi (persistent) strukturami danych.
- Potrafić zaprojektować rozwiązania problemów obliczeniowych, w szczególności różniczkowania numerycznego, z wykorzystaniem Lispa i jego funkcyjnego charakteru.
- Sprawnie posługiwać się metodami symbolicznego przetwarzania danych, w szczególności – realizować mechanizm różniczkowania symbolicznego wyrażeń algebraicznych w zakresie omówionym w literaturze podstawowej do przedmiotu.
- Sprawnie posługiwać się rekurencją jako metodą realizacji procesów obliczeniowych,wiedzieć, co to są procesy iteracyjne i rekurencyjne oraz jaki jest ich związek z procedurami definiowanymi w sposób rekurencyjny.
- Zdawać sobie sprawę z ograniczeń imperatywnych sposobów formułowania algorytmów.

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Harold Abelson, Julie Sussman, Gerald J. Sussman, Struktura i interpretacja programów komputerowych, Wydawnictwa Naukowo Techniczne , Wrzesień 2002, ISBN: 83-204-2712-6,

Literatura uzupełniająca przedmiotu:

— Michael Fogus, The Joy of Clojure: Thinking the Clojure Way [Paperback], Manning Publications; 1 edition (April 4, 2011), ISBN-10: 1935182641, ISBN-13: 978-1935182641 — Stuart Halloway, Programming Clojure (Pragmatic Programmers), Pragmatic Bookshelf; 1 edition (June 4, 2009), ISBN-10: 1934356336, ISBN-13: 978-193435633
Inne materiały dydaktyczne:
— Strony internetowe technologii: http://clojure.org/documentation — Podręcznik: http://en.wikibooks.org/wiki/Clojure_Programming

BAZY DANYCH I APLIKACJE

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) BAZY DANYCH I APLIKACJE			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy obowiązkowy	
Rok: 3	**Semestr: 6**	**ECTS ogółem:4**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 2**	**studia niestacjonarne: 2**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Bazy danych** **Wybrane środowiska programowania** **Języki i paradygmaty programowania**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: egzamin** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Konrad Grzanek**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	15	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	20
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	2
RAZEM:	**47**	**RAZEM:**	**32**
Praca własna studenta (PWS):	53	Praca własna studenta (PWS):	68
RAZEM z PWS:	**100**	**RAZEM z PWS:**	**100**
Sumaryczne obciążenie pracą studenta wg form aktywności:			

Forma aktywności:	Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**47**	**32**
2. Przygotowanie się do zajęć	13	28
3. Przygotowanie esejów		
4. Wykonanie projektów	20	20
5. Zapoznanie z literaturą podstawową	20	20
6. Pisemna praca zaliczeniowa		
7. Inne:		
SUMA:	**100**	**100**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Przedstawienie aspektów użycia technologii JDBC do budowy złożonych, transakcyjnych systemów opartych o relacyjne bazy danych. Techniki zapewnienia bezpieczeństwa transakcyjnego na poziomie aplikacji współpracującym z bazą danych.
2. Omówienie czynników wpływających na wydajność bazy danych – na przykładzie systemu Postresql. Strojenie Postgresa, aspekty sprzętowe i konfiguracyjne.
3. Wprowadzenie do tematyki nierelacyjnych baz danych – przegląd istniejących podejść oraz ich implementacji. Framework MapReduce oraz BigTable firmy Google.
4. System Berkeley DB Java Edition jako przykład silnika key→value store – wprowadzenie, instalacja i konfiguracja.
5. Omówienie API systemu Berkeley DB JE; składowanie obiektów, iteracja, aktualizacja składnicy, kursory.
6. Berkeley DB JE – trwałe kolekcje w Javie, omówienie użycia biblioteki z językiem Clojure – studium przypadku.
7. Memcached – wprowadzenie, instalacja i konfiguracja.
8. Użycie technologii memcached do optymalizacji działania aplikacji WEB.
9. Przeszukiwanie pełnotekstowe w relacyjnych bazach danych – przegląd istniejących rozwiązań.
10. Silnik indeksujący dokumenty tekstowe Lucene – wprowadzenie.
11. Omówienie API Lucene dla języka Java. Najważniejsze aspekty konfiguracji.
12. Podsumowanie wykładu.

LABORATORIA:

Celem projektów laboratoryjnych powinno być zastosowanie w praktyce technik implementacji systemów oprogramowania wykorzystujących poznane w trakcie wykładów rozwiązania bazodanowe. Z uwagi na stosunkowo dużą złożoność zadań, sugeruje się możliwość przeprowadzenia całosemestralnych projektów, w których realizacji mogą uczestniczyć 2 lub 3 osobowe zespoły studentów.

Przykładowe tematy do opracowania w trakcie zajęć laboratoryjnych:

1. Implementacja systemu zarządzania trwałymi obiektami języka Java z wykorzystaniem biblioteki Berkeley DB JE.
2. Optymalizacja aplikacji WEB wykonanej w technologii JEE z wykorzystaniem memcached – keszowanie wyrenderowanych stron, keszowanie wyników zapytań do relacyjnej bazy danych (Postgresql).
3. Wykorzystanie silnika Lucene do budowy systemu indeksującego dokumenty tekstowe oraz pełnotekstowej wyszukiwarki.
4. Realizacja transakcyjnego systemu opartego o relacyjną bazę danych Postgresql z wykorzystaniem języka

programowania wysokiego poziomu (sugerowana technologia JEE oraz standard JDBC).			
IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA			
Efekty kształcenia:			
Wiedza:			
Kod wg KRK:		**Kod KEK/ % udział przedmiotu w osiągnięciu efektu:**	**Metoda (forma) weryfikacji***
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Test wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych,inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego
Umiejętności:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	obserwacja i ocena wykonania zadania praktycznego
T1A_U09 T1A_U12 T1A_U16 InzA_U02 InzA_U04 InzA_U08	projektuje nieskomplikowane systemy baz danych wykorzystując przynajmniej jeden z powszechnie używanych systemów zarządzania bazami danych	K_U11/50%	obserwacja i ocena wykonania zadania praktycznego
Kompetencje społeczne:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań
* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?			
Cele kształcenia (przedmiotowe efekty kształcenia): Celem przedmiotu jest zapoznanie studenta z różnymi aspektami konstrukcji, konfiguracji i użycia rozwiązań			

bazodanowych, tak relacyjnych, jak i nierelacyjnych. Po zakończeniu kursu student:

- ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie baz danych relacyjnych i nierelacyjnych, implementujących język SQL oraz nie posiadających tej cechy (tzw. bazy NoSQL), składujących dane na trwałych nośnikach oraz w pamięci operacyjnej
- zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych opartych o bazy danych, relacyjne i nierelacyjne
- ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jeden z powszechnie używanych środowisk baz danych; potrafi ocenić złożoność obliczeniową algorytmów bazodanowych, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów

W szczególności student powinien:

- swobodnie posługiwać się relacyjnymi bazami danych z poziomu zorientowanego obiektowo języka programowania
- stosować standard JDBC do tworzenia transakcyjnych systemów informatycznych
- mieć świadomość istnienia czynników wpływających na wydajność działania relacyjnych systemów baz danych, znać podstawowe techniki analizy i optymalizacji dla silnika Postgresql
- znać najważniejsze elementy architektury systemu BigTable oraz algorytmu MapReduce i jego implementacji
- stosować bibliotekę Berkeley DB Java Edition do implementacji wydajnych systemów zarządzania danymi
- posiadać umiejętność konfiguracji i użycia mechanizmu memcached do optymalizacji szybkości działania portali internetowych i innych aplikacji WEB, w tym – do optymalizacji działania algorytmów współpracujących z bazami danych
- znać najważniejsze zagadnienia algorytmiczne związane z wyszukiwaniem pełnotekstowym oraz bazami danych realizującymi funkcje wyszukiwania tekstów
- potrafić używać silnik Lucene do konstrukcji systemów realizujących funkcjonalność indekserów i wyszukiwarek pełnotekstowych dla wybranych rodzajów dokumentów

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Gary Cornell, Cay Horstmann, Java. Techniki zaawansowane. Wydanie VIII, Helion 2009
— Documentation: Berkeley DB Java Edition, http://download.oracle.com/docs/cd/E17277_02/html/index.html
— Data sheet: Oracle Berkeley DB Java Edition, http://www.oracle.com/technetwork/products/berkeleydb/berkeley-db-je-ds-066564.pdf
— J. Finsel, Using memcached. How to scale your website easily, The Pragmatic Bookshelf 2008
— Hatcher E., Gospodnetic O., McCandless M., 2010, Lucene in Action, Second Edition, Manning Publications, ISBN: 1933988177

Literatura uzupełniająca przedmiotu:

— J.D. Ullman, J. Widom, Podstawowy wykład z systemów baz danych, WNT, W-wa, 2000 (seria: Klasyka Informatyki)

Inne materiały dydaktyczne:

— Chang, Fay; Dean, Jeffrey; Ghemawat, Sanjay; Hsieh, Wilson C; Wallach, Deborah A; Burrows, Michael ‘Mike’; Chandra, Tushar; Fikes, Andrew (2006), "Bigtable: A Distributed Storage System for Structured Data" (PDF), Research, Google.
— http://wiki.postgresql.org/wiki/Performance_Optimization

SYSTEMY SZKIELETOWE (JEE)

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) SYSTEMY SZKIELETOWE (JEE)			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy obowiązkowy	
Rok: 3	**Semestr: 6**	**ECTS ogółem:4**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 2**	**studia niestacjonarne: 1**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Wybrane środowiska programowania** **Języki i paradygmaty programowania** **Programowanie współbieżne** **Wzorce architektoniczne oprogramowania w biznesie**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: zaliczenie** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Konrad Grzanek**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	15	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	10
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	2
RAZEM:	**47**	**RAZEM:**	**22**
Praca własna studenta (PWS):	53	Praca własna studenta (PWS):	78

RAZEM z PWS:	100	RAZEM z PWS:	100

Sumaryczne obciążenie pracą studenta wg form aktywności:

Forma aktywności:	Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**47**	**22**
2. Przygotowanie się do zajęć	18	38
3. Przygotowanie esejów		
4. Wykonanie projektów	20	20
5. Zapoznanie z literaturą podstawową	15	20
6. Pisemna praca zaliczeniowa		
7. Inne:		
SUMA:	**100**	**100**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY

1. Technologia JEE, historia, najważniejsze komponenty, zagadnienia architektoniczne, zastosowania.
2. Testowanie oprogramowania tworzonego w języku Java w kontekście użycia platformy JEE, wprowadzenie do użycia biblioteki JUnit
3. Techniki projektowania oraz standardy kodowania na platformie JEE.
4. Wzorzec projektowy Dependency Injection (IoC), standard JNDI.
5. Technologie dostępu do danych, w tym – do relacyjnych baz danych – na platformie JEE, standard JDBC.
6. Komponenty Enterprise Java Beans, stanowość i bezstanowość, ziarna sesyjne.
7. Warstwa WEB w modelu MVC. Servlety i strony JSP.
8. Projektowanie kontrolerów w warstwie WEB. Sprzęgi z komponentami EJB.
9. Aspekty implementacyjne na bazie serwera aplikacyjnego Resin.
10. Dystrybucja oprogramowania w technologii JEE, aplikacje WEB, aplikacje klasy enterprise, pakowanie, narzędzia dystrybucyjne.
11. Przegląd najważniejszych rozwiązań sprzętowych i systemowych służących do budowania aplikacji w technologii JEE.
12. Podsumowanie.

LABORATORIA

Studenci pod kierownictwem prowadzącego wykonują projekt działającego systemu biznesowego w oparciu o platformę JEE. Poszczególne jego etapy odpowiadają treściom wykładowym, a powstanie całości rozłożone jest na cały semestr. Proponowany przebieg:

1. Wybór tematu projektu, konsultacja z prowadzącym aspektów architektury rozwiązania i jego implementacji.
2. Implementacja bazy danych oraz komponentów EJB odpowiedzialnych za dostęp do danych.
3. Implementacja kontrolerów sprzęgających warstwę modelu z warstwą WEB.
4. Implementacja warstwy WEB.
5. Testowanie aplikacji i przygotowanie jej wersji dystrybucyjnej.

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:			
Wiedza:			
Kod wg KRK:		**Kod KEK/%: udział przedmiotu w osiągnięciu efektu:**	**Metoda (forma) weryfikacji***
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego
Umiejętności:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	obserwacja i ocena wykonania zadania praktycznego
Kompetencje społeczne:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań
* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?			

Cele kształcenia (przedmiotowe efekty kształcenia):

Celem kursu jest zapoznanie studenta z platformą JEE, ze specyfiką projektowania i budowania aplikacji w oparciu o tę platformę, z najważniejszymi komponentami, dobrymi praktykami, metodami testowania oraz sposobami dystrybuowania aplikacji.

Po ukończeniu kursu student:

- ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie posługiwania się platformą Java Enterprise Edition w celu budowania zaawansowanych systemów informatycznych
- zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych z wykorzystaniem technologii JEE, potrafi tworzyć sprzęgi tego rodzaju aplikacji z zewnętrznymi źródłami danych
- ma umiejętność formułowania algorytmów i ich implementacji stosując środowiska programistyczne

właściwe dla technologii JEE

W szczególności student:

- zna najważniejsze założenia, technologie składowe, zagadnienia związane z architekturą oraz zastosowania technologii JEE
- potrafi tworzyć testy jednostkowe dla komponentów technologii JEE, posługuje się swobodnie biblioteką Junit
- zna techniki projektowania i implementacji właściwe dla tej platformy, ma świadomość istnienia standardów kodowania w języku Java
- potrafi posługiwać się komponentami podlegającymi zarządzaniu przez kontenery implementujące wzorzec Dependency Injection
- zna standard JNDI, rozumie jego zalety i ograniczenia, potrafi zarządzać konfiguracjami komponentów w tym standardzie
- posługuje się komponentami EJB w zakresie dostępu do danych (mapping obiektowo-relacyjny) oraz utrzymania sesji
- posiada podstawowe umiejętności wykorzystania servletów oraz stron JSP do implementowania widoków w warstwie WEB aplikacji
- potrafi przygotowywać archiwa dystrybucyjne oraz instalować oprogramowanie tworzone w technologii JEE

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Rod Johnson, Expert One on One J2ee Design and Development, Peer Information 2002
— Adam Bochenek, Prosty przepis na J2EE: Boss, Eclipse i komponenty EJB, PWN 2009

Literatura uzupełniająca przedmiotu:

— Rod Johson, Expert One-on-One J2EE Development without EJB, Wrox 1st edition 2004

Inne materiały dydaktyczne:

— http://www.oracle.com/technetwork/java/javaee/blueprints/index.html
— http://www.caucho.com/
— http://www.caucho.com/documentation-resin-app-server/

PROGRAMOWANIE I KONFIGURACJA SERWERÓW APLIKACYJNYCH

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) PROGRAMOWANIE I KONFIGURACJA SERWERÓW APLIKACYJNYCH			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy obowiązkowy	
Rok: 3	**Semestr: 6**	**ECTS ogółem:2**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Wybrane środowiska programowania** **Języki i paradygmaty programowania** **Programowanie współbieżne** **Wzorce architektoniczne oprogramowania w biznesie**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: zaliczenie** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Konrad Grzanek**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	10	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	10	Laboratorium:	10
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	2
RAZEM:	**22**	**RAZEM:**	**22**
Praca własna studenta (PWS):	28	Praca własna studenta (PWS):	28
RAZEM z PWS:	**50**	**RAZEM z PWS:**	**50**

Sumaryczne obciążenie pracą studenta wg form aktywności:		
Forma aktywności:	**Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:**	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**22**	**22**
2. Przygotowanie się do zajęć	8	8
3. Przygotowanie esejów		
4. Wykonanie projektów	10	10
5. Zapoznanie z literaturą podstawową	10	10
6. Pisemna praca zaliczeniowa		
7. Inne:		
SUMA:	**50**	**50**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Przegląd platformy JEE, przedstawienie najważniejszych serwerów aplikacyjnych zgodnych ze standardem JEE.
2. Wydajność aplikacji tworzonych w języku Java: przedstawienie aspektów strojenia wydajności (performance tuning).
3. Omówienie zasad działania mechanizmów automatycznego zarządzania pamięcią (Garbage Collection) na platformie Java. Wpływ wyboru mechanizmów GC na wydajność aplikacji w języku Java.
4. Omówienie architektury serwera aplikacyjnego na przykładzie serwera Resin 4.
5. Czynności administracyjne i konfiguracyjne serwera Resin.
6. Konfiguracja aplikacji JEE na serwerze Resin.
7. Instalacja i konfiguracja serwera aplikacyjnego JBoss – wprowadzenie.
8. Wybrane aspekty administracji serwerem aplikacyjnym JBoss.
9. Użycie mechanizmów rozszerzeń maszyny wirtualnej Javy do monitorowania stanu działającej aplikacji JEE.
10. Instalacja rozszerzeń znanych zintegrowanych środowisk programistycznych (IDE) pozwalających na zarządzanie serwerami aplikacyjnymi.ących

LABORATORIA:

1. Instalacja i konfiguracja serwera aplikacyjnego Resin.
2. Instalacja i konfiguracja serwera aplikacyjnego JBoss.
3. Wybór istniejącej aplikacji WEB. Instalacja i konfiguracja bazy danych dla aplikacji. Instalacja aplikacji w środowiskach Resin i JBoss, jej uruchomienie i obserwacja działania przy użyciu istniejących narzędzi monitorujących.
4. Instalacja rozszerzenia wybranego środowiska programistycznego (np. Eclipse) do współpracy z serwerem aplikacyjnym Resin lub JBoss.
5. Strojenie wydajności wybranej aplikacji WEB.

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:

Wiedza:

Kod wg KRK:		**Kod KEK/%: udział**	**Metoda (forma)**

		przedmiotu w osiągnięciu efektu:	weryfikacji*
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Test wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego
Umiejętności:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	obserwacja i ocena wykonania zadania praktycznego
Kompetencje społeczne:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań

* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?

Cele kształcenia (przedmiotowe efekty kształcenia):

Podstawowym celem kształcenia jest zapoznanie studenta z najważniejszymi aspektami instalacji, konfiguracji i administracji serwerami aplikacyjnymi zgodnymi ze standardem JEE.

Po zakończeniu kursu student:

- ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie instalacji, konfiguracji i administracji serwerów aplikacyjnych, jak również w zakresie strojenia wydajności serwera oraz osadzonych w nim aplikacji
- zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu zadań informatycznych z zakresu zarządzania serwerami aplikacyjnymi zgodnymi ze standardem JEE oraz strojenia wydajności aplikacji, w tym - aplikacji WEB - w nich osadzonych
- ma umiejętność implementacji aplikacji zgodnych ze standardem JEE stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych wyposażonych w mechanizmy integracji z serwerami aplikacyjnymi

W szczególności student powinien:

- znać najważniejsze serwery aplikacyjne zgodne ze standardem JEE

- potrafić wpływać na parametry uruchomieniowe maszyny wirtualnej Javy w celu osiągania celów związanych z wydajnością aplikacji
- rozumieć działanie mechanizmów automatycznego zarządzania pamięcią w Javie i potrafić dobrać rodzaj mechanizmu GC do profilu aplikacji
- umieć posługiwać się wybranymi narzędziami służącymi do monitorowania stanu serwera aplikacyjnego, potrafić analizować logi serwera
- potrafić stroić wydajność aplikacji osadzonej w serwerze, określać charakterystykę jej tworzenia i usuwania obiektów
- znać rozszerzenia znanych środowisk programistycznych pozwalające na zarządzanie serwerami aplikacyjnymi oraz na realizację osadzania w nich tworzonch aplikacji
- potrafić zainstalować i zarządzać serwerem aplikacyjnym Resin
- potrafić zainstalować i zarządzać serwerem aplikacyjnym JBoss

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Adam Bochenek, Prosty przepis na J2EE: Boss, Eclipse i komponenty EJB, PWN 2009
— Richard Hightower, Joseph D. Gradecki, Mastering Resin, Wiley 1st edition (August 15, 2003)
— Jack Shirazi, Java Performance Tuning, 2nd Edition, O'Reilly 2003

Literatura uzupełniająca przedmiotu:

— Javid Jamae, Peter Johnson, JBoss in Action. Configuring the JBoss Application Server. Manning Publications Co. 2009, ISBN: 1933988029

Inne materiały dydaktyczne:

— http://www.caucho.com/
— http://www.caucho.com/documentation-resin-app-server/

MAPPERY OBIEKTOWO-RELACYJNE (HIBERNATE)

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) MAPPERY OBIEKTOWO-RELACYJNE (HIBERNATE)			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy obowiązkowy	
Rok: 4	**Semestr: 7**	**ECTS ogółem:4**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 2**	**studia niestacjonarne: 1**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Bazy danych i aplikacje** **Wybrane środowiska programowania** **Języki i paradygmaty programowania** **Systemy szkieletowe JEE**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: egzamin** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Konrad Grzanek**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	15	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	10
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	3
RAZEM:	**47**	**RAZEM:**	**23**
Praca własna studenta (PWS):	53	Praca własna studenta (PWS):	77
RAZEM z PWS:	**100**	**RAZEM z PWS:**	**100**

Sumaryczne obciążenie pracą studenta wg form aktywności:		
Forma aktywności:	**Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:**	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**47**	**23**
2. Przygotowanie się do zajęć	13	28
3. Przygotowanie esejów		
4. Wykonanie projektów	20	20
5. Zapoznanie z literaturą podstawową	20	29
6. Pisemna praca zaliczeniowa		
7. Inne:		
SUMA:	**100**	**100**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Omówienie najważniejszych zagadnień związanych z tworzeniem i zarządzaniem trwałymi obiektami w języku Java. Niezgodność paradygmatów, serializacja, obiektowe języki zapytań do baz danych, mapowanie obiektowo-relacyjne.
2. Instalacja i konfiguracja mappera obiektowo-relacyjnego Hibernate do współpracy z serwerem aplikacyjnym.
3. Modele dziedzinowe, ich implementacja i meta-konfiguracja mechanizmów mapowania obiektowo-relacyjnego.
4. Klasy trwałych obiektów, encje i typy, opcje mapowania.
5. Dziedziczenie i hierarchie typów w ujęciu Hibernate, różne sposoby odwzorowania hierarchii trwałych klas na model relacyjny, rozważania o wydajności.
6. Mapowanie kolekcji i związków pomiędzy obiektami, kolekcje z adnotacjami, mapowanie związków rodzic-dziecko.
7. Zaawansowane mapowanie związków (asocjacji); asocjacje polimorficzne.
8. Integracja spadkowych (ang. legacy) systemów baz danych z mapperem.
9. Działania na obiektach trwałych, cykl życia, tożsamość obiektów, interfejsy i API Hibernate, Java Persistence API a Hibernate, używanie mechanizmów Java Persistence w komponentach EJB.
10. Hibernate a transakcyjność, mechanizmy optymalizacji dostępu transakcyjnego do danych przy użyciu mappera.
11. Zarządzanie sesjami Hibernate, propagowanie sesji w wątku lokalnym, poprzez JTA oraz z wykorzystaniem EJB.
12. Metody szybkiego modyfikowania stanu trwałych obiektów.
13. Optymalizacja wydobywania trwałych obiektów z bazy, przetwarzanie wsadowe, keszowanie wyników operacji.
14. Zapytania HQL oraz Java Persistence API Query Language.
15. Testowanie aplikacji opartych o Hibernate, podsumowanie wykładu.

LABORATORIA:

Preferowaną formą zaliczenia laboratoriów jest zespołowa (2-3 osoby) realizacja całosemestralnego projektu polegającego na utworzeniu wybranej aplikacji biznesowej z wykorzystaniem mappera obiektowo-relacyjnego Hibernate. Studenci powinni posłużyć się technologiami wskazanymi przez prowadzącego, te z kolei powinny odpowiadać treściom wykładowym.

Studenci dzielą się elementami funkcjonalnymi do wykonania a ich ocena końcowa zależy zarówno od indywidualnych osiągnięć w realizacji przydzielonych im zadań, jak i od całokształtu realizowanego zespołowo projektu.

Prowadzący zajęcia laboratoryjne powinien wspierać studentów swoimi uwagami i pomagać w rozwiązywaniu najtrudniejszych problemów. Powinien również dokonywać okresowej kontroli osiągnięć, tak całego zespołu, jak i poszczególnych jego członków.

Szczególna uwaga powinna być położona na następujące aspekty mechanizmów mappingu:

- Mapowanie związków pomiędzy obiektami trwałymi
- Opracowywanie modeli dziedzinowych sprzyjających dobrej wydajności i skalowalności aplikacji
- Odwzorowanie hierarchii klas
- Prawidłowa realizacja mechanizmów transakcyjnych

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:

Wiedza:

Kod wg KRK:		**Kod KEK/%: udział przedmiotu w osiągnięciu efektu:**	**Metoda (forma) weryfikacji***
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Test wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych,inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego
Umiejętności:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego
Kompetencje społeczne:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań

* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub

projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?

Cele kształcenia (przedmiotowe efekty kształcenia):

W przebiegu przedmiotu studenci mają nabyć umiejętności w zakresie posługiwania się mapperem obiektowo relacyjnym Hibernate w procesie tworzenia zaawansowanych systemów biznesowych wykorzystujących relacyjne baz danych oraz inne mechanizmy trwałego składowania danych.

Po ukończeniu kursu student:

- ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie posługiwania się mechanizmami mappera obiektowo-relacyjnego Hibernate, zna najważniesze zagadnienia odzorowania obiektów w zorientowanym obiektowo języku programowania Java na modele trwałych danych
- zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych zarządzających danymi z wykorzystaniem algorytmów realizujących logikę biznesową korzystających z mappera obiektowo-relacyjnego
- ma umiejętność formułowania algorytmów biznesowych i ich implementacji stosując mapper obiektowo-relacyjny Hibernate; potrafi ocenić złożoność obliczeniową tych algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów

W szczególności student powinien:

- rozumieć najważniejsze aspekty mariażu obiektowych technologii programowania oraz mechanizmów trwałego składowania informacji, znać w podstawowym zakresie serializację obiektów Javy, rozumieć znaczenie mappingu obiektowo-relacyjnego z punktu widzenia rozwiązania problemu niezgodności „impedancji" (ang. impedance mismatch) pomiędzy zorientowanymi obiektowo językami programowania a relacyjnym modelem danych
- potrafić zainstalować i skonfigurować Hibernate do celów związanych z budowaniem aplikacji na platformie JEE
- umieć tworzyć klasy trwałych obiektów, konfigurować je, budować hierarchie trwałych klas
- znać metody odwzorowania związków (association) pomiędzy trwałymi obiektami, w tym związków wiele-do jeden, wiele-do-wielu, związków polimorficznych
- potrafić użyć mapper Hibernate w połączeniu z istniejącą, spadkową bazą danych
- znać cykl życia trwałych obiektów, potrafić określić wpływ tego cyklu życia na wydajność aplikacji
- znać Java Persistence API, potrafić wykorzystać mapper Hibernate w połączeniu z technologią EJB
- potrafić stosować mechanizmy transakcyjności, optymalizować ich użycie
- znać metody optymalnego przetwarzania większych zbiorów danych (trwałych obiektów), stosować mechanizmy keszowania
- umieć posługiwać się językiem zapytań HQL oraz językiem zapytań JPA (Java Persistence API)
- potrafić tworzyć testy dla algorytmów wykorzystujących funkcjonalność mappera Hibernate

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Christian Bauer, Gavin King, Hibernate w akcji, Helion, Lipiec 2007, ISBN: 978-83-246-0527-9
— Christian Bauer, Gavin King, Java Persistence with Hibernate, Manning Publications; Revised edition (November 24, 2006), ISBN-10: 1932394885, ISBN-13: 978-1932394887

Literatura uzupełniająca przedmiotu:

— Paul Tepper Fisher, Brian D. Murphy, Spring Persistence with Hibernate (Beginning), Apress; 1 edition (November 2, 2010), ISBN-10: 1430226323, ISBN-13: 978-1430226321

Inne materiały dydaktyczne:

— Hibernate Reference Documentation: http://docs.jboss.org/hibernate/stable/core/manual/en-US/html/

BIZNESOWE SYSTEMY COTS (COMMERCIAL OFF-THE-SHELF)

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) BIZNESOWE SYSTEMY COTS (*Commercial off-the-shelf*)			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy obowiązkowy	
Rok: 4	**Semestr: 7**	**ECTS ogółem:4**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 2**	**studia niestacjonarne: 1**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Wzorce architektoniczne oprogramowania w biznesie** **Bazy danych i aplikacje**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: zaliczenie** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Konrad Grzanek**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	15	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	10
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	3
RAZEM:	**47**	**RAZEM:**	**23**
Praca własna studenta (PWS):	53	Praca własna studenta (PWS):	77
RAZEM z PWS:	**100**	**RAZEM z PWS:**	**100**
Sumaryczne obciążenie pracą studenta wg form aktywności:			
Forma aktywności:		**Szacowana liczba godzin potrzebnych**	

	na zrealizowanie aktywności:	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**47**	**23**
2. Przygotowanie się do zajęć	13	28
3. Przygotowanie esejów		
4. Wykonanie projektów	20	20
5. Zapoznanie z literaturą podstawową	20	29
6. Pisemna praca zaliczeniowa		
7. Inne:		
SUMA:	**100**	**100**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Przegląd istniejących rozwiązań biznesowych, moduły E-commerce firmy ATG, Tibco, Oracle Business Solutions, SAP i systemy ERP.
2. Wprowadzenie do systemu Open ERP, przedstawienie architektury i najważniejszych modułów.
3. Instalacja i konfiguracja Open ERP w systemie Linux.
4. Instalacja i konfiguracja Open ERP w systemie Windows.
5. Interfejsy systemu Open ERP – konfiguracja i użycie.
6. Sprzedaż i zamówienia – zarządzanie relacjami z klientami.
7. Sprzedaż i zamówienia – helpdesk, zarządzanie relacjami z dostawcami, profiowanie, narzędzia komunikacyjne.
8. Faktury i płatności – zarządzanie przepływem faktur.
9. Faktury i płatności – zarządzanie rachunkowością firmy i automatyczne tworzenie faktur.
10. Faktury i płatności – zarządzanie płatnościami.
11. Analizy finansowe, raportowanie, zarządzanie opodatkowaniem ustawowym.
12. Zarządzanie zasobami ludzkimi
13. Zarządzanie procesem produkcyjnym.
14. Zarządzanie dokumentami i procesami, użycie kalendarzy.
15. Zaawansowana konfiguracja i administracja.

LABORATORIA:

Prefererowaną formą zaliczenia laboratoriów jest zespołowa (2-3 osoby) realizacja całosemestralnego projektu polegającego na utworzeniu systemu ERP dla przykładowego przedsiębiorstwa.

Studenci dzielą się elementami funkcjonalnymi do wykonania a ich ocena końcowa zależy zarówno od indywidualnych osiągnięć w realizacji przydzielonych im zadań, jak i od całokształtu realizowanego zespołowo projektu.

Opracowane elementy biznesowe powinny odpowiadać cechom systemu Open ERP prezentowanym w trakcie wykładów.

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:

Wiedza:

Kod wg KRK:		**Kod KEK/%: udział**	**Metoda (forma) weryfikacji***

		przedmiotu w osiągnięciu efektu:	
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Test wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych,inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego
Umiejętności:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego
Kompetencje społeczne:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań

* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?

Cele kształcenia (przedmiotowe efekty kształcenia):

Celem kształcenia jest zapoznanie studentów z gotowymi, komercyjnymi lub dystrybuowanymi za darmo systemami wspierającymi działalność przedsiębiorstw. Przykładowym systemem, na którym koncentruje się uwaga, jest system zarządzania zasobami przedsiębiorstw Open ERP.

Po zakończeniu kursu student:

- ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie wykorzystania biznesowych systemów COTS (ang. Commercial off-the-shelf) do usprawnienia działania wybranych obszarów działalności przedsiębiorstw, szczególnie z wykorzystaniem oprogramowania Open ERP
- zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu zadań informatycznych z zakresu analizy, projektowania i budowy systemów wspierających działalność przedsiębiorstw z wykorzystaniem gotowych komponentów oprogramowania dostępnych na rynku IT
- ma umiejętność dostrzegania, opizywania i formalizowania problemów biznesowych oraz implementacji ich rozwiązań z wykorzystaniem gotowych rozwiązań biznesowych dostępnych komercyjnie lub osiągalnych bez konieczności ponoszenia kosztów związanych z ich zakupem

W szczególności student powinien:

- znać najważniejsze dostępne na rynku rozwiązania biznesowe wspierające działalność przedsiębiorstw
- potrafić zidentyfikować problemy biznesowe przedsiębiorstwa związane z zarządzaniem informacją, umie dobrać rozwiązanie do zidentyfikowanego problemu, oszacować jego koszt i zidentyfikować najważniejsze problemy związane z wdrożeniem
- znać architekturę systemu Open ERP, jego cechy, powinien również być zdolny do dokonania analizy porównawczej systemu Open ERP z istejącymi rozwiązaniami klasy ERP
- posiadać umiejętność zainstalowania systemu Open ERP w systemach operacyjnych Linux oraz Windows
- posiadać umiejętności w zakresie administrowania i utrzymania systemów opartych o Open ERP
- potrafić wykorzystywać moduły Open ERP związane ze sprzedażą, zamówieniami i zarządzaniem płatnościami
- umieć posłużyć się danymi zgromadzonymi w systemie w celu generowania raportów i predykcji
- umieć zarządzać zasobami ludzkimi w oparciu o odpowiednie moduły Open ERP
- znać zagadnienia związane z zarządzaniem procesem produkcyjnym

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Open ERP 6, a modern approach to integrated business management, (http://v6.openerp.com/services/books)
— Open Object Installation Manuals, (http://v6.openerp.com/services/books)
— Fabien Pinckaers, Geoff Gardiner, Open ERP for Retail and Industrial Management, Tiny SPRL (2009), ISBN-10: 2960087607, ISBN-13: 978-2960087604

Literatura uzupełniająca przedmiotu:

— Open Object Business Intelligence, (http://v6.openerp.com/services/books)
— Open ERP Features, (http://v6.openerp.com/services/books)
— Open Object Community Book, (http://v6.openerp.com/services/books)
— Open Object Developer Book, (http://v6.openerp.com/services/books)
— Open Object Technical Guide, (http://v6.openerp.com/services/books)

Inne materiały dydaktyczne:

TECHNOLOGIE PROGRAMOWANIA (MODUŁY SPECJALNOŚCIOWE DO WYBORU)

PROJEKTOWANIE I PROGRAMOWANIE Z WYKORZYSTANIEM WZORCÓW PROJEKTOWYCH

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) PROJEKTOWANIE I PROGRAMOWANIE Z WYKORZYSTANIEM WZORCÓW PROJEKTOWYCH			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy fakultatywny	
Rok: 3	**Semestr: 5**	**ECTS ogółem:4**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 2**	**studia niestacjonarne: 1**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Podstawy programowania, Wybrane środowiska programowania, Języki i paradygmaty programowania**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi.** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: egzamin ustny** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Zbigniew Filutowicz**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	15	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	10
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	2
RAZEM:	**47**	**RAZEM:**	**22**

Praca własna studenta (PWS):	53	Praca własna studenta (PWS):	78
RAZEM z PWS:	**100**	**RAZEM z PWS:**	**100**

Sumaryczne obciążenie pracą studenta wg form aktywności:

Forma aktywności:	**Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:**	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**47**	**22**
2. Przygotowanie się do zajęć	23	38
3. Przygotowanie esejów	5	5
4. Wykonanie projektów	10	10
5. Zapoznanie z literaturą podstawową	10	20
6. Pisemna praca zaliczeniowa	5	5
7. Inne:		
SUMA:	**100**	**100**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Pojęcia dobrej praktyki inżynierskiej, paradygmatu oraz wzorca projektowego. Wzorce projektowe w sztuce, socjologii oraz informatyce. Wzorce architektoniczne w informatyce.
2. Inżynieria oprogramowania, paradygmaty projektowania i programowania, programowanie OOP. Analiza potrzeb, projektowanie i programowanie.
3. Klasyfikacja wzorców projektowych: czynnościowe, konstrukcyjno-kreacyjne, odwzorowania O-R, strukturalne i współbieżności.
4. Inne rodzaje wzorców projektowych np. workflow
5. Wady i zalety stosowania wzorców projektowych, anty-wzorce projektowy.
6. Wzorzec projektowy MVC Model-View-Controller (Model-Widok-Kontroler)
7. Inżynieria oprogramowania i wzorce projektowe. Narzędzie programistyczne typu IDE i CASE.
8. Przykładowe implementacje wzorców projektowych w językach programowania.

LABORATORIA

1. Organizacja zajęć. Omówienie zasad wykonania projektów oraz zasad zaliczenia zajęć. Wprowadzenie w tematykę przedmiotu. Omówienie zasobów wiedzy bibliograficznej i netograficznej.
2. Wybór przykładowych wzorców projektowych, analiza ich zastosowań na etapie projektowania i implementacja w wybranych językach programowania – opracowanie projektu i jego dokumentacja.
3. Wybór przykładowych antywzorców projektowych, analiza ich zastosowań na etapie projektowania i implementacja w wybranych językach programowania – opracowanie projektu i jego dokumentacja.
4. Wspólne referowanie opracowanych projektów i dyskusja uzyskanych wyników oraz wnioski

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:

Wiedza:

Kod wg KRK:		**Kod KEK/%: udział przedmiotu w**	**Metoda (forma) weryfikacji***

		osiągnięciu efektu:	
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Egzamin, ustny sprawdzian wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Egzamin, ustny sprawdzian wiedzy
Umiejętności:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	Ocena zadań projektowych oraz obserwacja wykonania zadań praktycznych
Kompetencje społeczne:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań

* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?

Cele kształcenia (przedmiotowe efekty kształcenia):

Celem zajęć jest przedstawienie podstawowych zagadnień i wyrobienie umiejętności z zakresu inżynierii oprogramowania opartej na wzorcach projektowych.

Na zajęciach laboratoryjnych studenci analizują przykładowe wykorzystania wzorców projektowych oraz zdobywają umiejętności ich stosowania ich w wybranych projektach informatycznych. Studenci implementują opracowane algorytmy przetwarzania danych z wykorzystaniem wzorców projektowych, pod nadzorem prowadzącego.

Po ukończeniu kursu student:

- Ma wiedzę z zakresu inżynierii oprogramowania opartej na wzorcach projektowych.
- Poznaje wybrane wzorce projektowe stosowanie w inżynierii oprogramowania i rozumie zasady ich używania.
- Poznaje wybrane antywzorce projektowe opisane w literaturze.
- Potrafi ocenić przydatność podstawowych wzorców projektowych w konkretnych rozwiązaniach.
- Potrafi używać wzorców projektowych do projektowania i programowania systemów oprogramowania.
- Potrafi wykorzystywać oprogramowanie do komputerowego wspomagania projektowania i

implementacji projektów informatycznych z wykorzystaniem środowisk programistycznych IDE i CASE.

- Ma umiejętność przekazywania wiedzy i dyskusji na profesjonalnym w zakresie inżynierii wzorców projektowych.

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Jenifer Tidwell, Projektowanie interfejsów. Sprawdzone wzorce projektowe. Helion 2012.
— Elisabeth Freeman, Eric Freeman, Bert Bates, Kathy Sierra, Wzorce projektowe. Rusz głową!. Helion 2010.
— Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: Inżynieria oprogramowania: Wzorce projektowe (Wyd. II). Warszawa: WNT, 2008

Literatura uzupełniająca przedmiotu:

— James William Cooper, Java. Wzorce projektowe, Helion 2001.
— Steven John Metsker, C#. Wzorce projektowe, Helion 2005.

Inne materiały dydaktyczne:

— Workflow patterns http://www.workflowpatterns.com/patterns/presentation/abstractsyntax/index.php
— Anti-patterns http://en.wikipedia.org/wiki/Category:Anti-patterns
— Software design patterns http://en.wikipedia.org/wiki/Category:Software_design_patterns

WZORCE ARCHITEKTONICZNE OPROGRAMOWANIA W BIZNESIE

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) WZORCE ARCHITEKTONICZNE OPROGRAMOWANIA W BIZNESIE			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy fakultatywny	
Rok: 3	**Semestr: 5**	**ECTS ogółem:4**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 2**	**studia niestacjonarne: 1**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Algorytmy i złożoność** **Języki i paradygmaty programowania**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: egzamin** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Konrad Grzanek**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	15	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	10
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	2
RAZEM:	**47**	**RAZEM:**	**22**
Praca własna studenta (PWS):	53	Praca własna studenta (PWS):	78
RAZEM z PWS:	**100**	**RAZEM z PWS:**	**100**
Sumaryczne obciążenie pracą studenta wg form aktywności:			

Forma aktywności:	Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	47	22
2. Przygotowanie się do zajęć	18	38
3. Przygotowanie esejów		
4. Wykonanie projektów	20	20
5. Zapoznanie z literaturą podstawową	15	20
6. Pisemna praca zaliczeniowa		
7. Inne:		
SUMA:	**100**	**100**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Pojęcie wzorca projektowego, znaczenie wzorców w procesie powstawania oprogramowania, dobre praktyki związane z procesem budowy systemów informatycznych.
2. Najważniejsze kryteria oceny jakości oprogramowania klasy enterprise: opóźnienie (latency), przepustowość (throughput), szybkość (performance), skalowalność, efektywność. Wpływ wymienionych czynników na proces powstawania systemów biznesowych. Analiza wybranych studiów przypadku.
3. Logika biznesowa – omówienie pojęcia. Podział oprogramowania biznesowego na warstwy. Rys historyczny: ewolucja architektur w aplikacjach klasy enterprise.
4. Platforma Java Enterprise Edition jako środowisko do tworzenia wielowarstwowych aplikacji biznesowych. Zalety i niedomagania. Wzorce na platformie JEE jako jeden ze sposobów eliminacji występujących problemów.
5. Przemyślenia projektowe dotyczące warstwy prezentacji, anty-wzorce warstwy prezentacji.
6. Wzorce warstwy prezentacji: Intercepting Filter, Front Controller, View Helper.
7. Wzorce warstwy prezentacji: Composite View, Service to Worker, Dispatcher View.
8. Przemyślenia projektowe dotyczące warstwy (logiki) biznesowej, anty-wzorce warstwy (logiki) biznesowej.
9. Wzorce warstwy (logiki) biznesowej: Business Delegate, Value Object, Session Facade, Composite Entity.
10. Wzorce warstwy (logiki) biznesowej: Value Object Assembler, Value List Handler, Service Locator.
11. Przemyślenia projektowe dotyczące warstwy integracji, anty-wzorce warstwy integracji.
12. Wzorce warstwy integracji: Data Access Object, Service Activator.
13. Obiekty rozproszone w aplikacjach klasy enterprise, omówienie technologii CORBA. Technologia RMI oraz je zastosowania. Kontenery komponentów Enterprise Java Beans.
14. Metody zapewniania bezpieczeństwa i efektywności działania mechanizmów transakcyjnego przetwarzania danych w systemach klasy enterprise.
15. Aspekty zarządzania rozproszoną sesją w aplikacjach biznesowych, znaczenie bezstanowości aplikacji. Studia przypadków na przykładzie dużych aplikacji WEB (Reddit, YouTube, Ebay).

LABORATORIA

Studenci pod kierownictwem prowadzącego wykonują projekt działającego systemu biznesowego w oparciu o platformę JEE. Poszczególne jego etapy odpowiadają treściom wykładowym, a powstanie całości rozłożone jest na cały semestr. Proponowany przebieg:

1. Wybór tematu projektu, konsultacja wyboru z prowadzącym, podstawowa dekompozycja oprogramowania na warstwy.
2. Implementacja modelu danych i warstwy logiki biznesowej
3. Implementacja warstwy prezentacji.
4. Implementacja mechanizmów transakcyjnych.
5. Implementacja wybranych komponentów integracyjnych i/lub rozproszonej sesji użytkownika.

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:

Wiedza:

Kod wg KRK:		**Kod KEK/%: udział przedmiotu w osiągnięciu efektu:**	**Metoda (forma) weryfikacji***
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Test wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego

Umiejętności:

Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	obserwacja i ocena wykonania zadania praktycznego

Kompetencje społeczne:

Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań

* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?

Cele kształcenia (przedmiotowe efekty kształcenia):

Celem kursu jest zapoznanie studenta z wykorzystaniem wzorców projektowych stosowanych w procesie tworzenia wielowarstwowych aplikacji biznesowych.

Po ukończeniu kursu student:

- ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie projektowania i implementowania wielowarstwowych aplikacji klasy enterprise
- zna podstawowe metody, techniki i narzędzia stosowane przy projektowaniu i implementowaniu wielowarstwowych aplikacji biznesowych, w szczególności warstw logiki biznesowej, warstwy prezentacji oraz warstw integracji z systemami zewnętrznymi
- ma umiejętność formułowania algorytmów związanych z logiką biznesową, mechanizmami transakcyjnymi, mechanizmami utrzymania rozproszonej sesji użytkownika, mechanizmami rozpraszania obiektów biznesowych, ma umiejętność ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową tych algorytmów, ich skalowalność i szybkość działania całego systemu, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów

W szczególności student powinien

- Znać pojęcie wzorca projektowego, znaczenie tego pojęcia w inżynierii oprogramowania, w szczególności - w procesie tworzenia wielowarstwowych aplikacji klasy enterprise
- Posługiwać się dobrymi praktykami związanymi z powstawaniem systemów biznesowych
- Znać najważniejsze kryteria oceny jakości oprogramowania, rozumieć wpływ architektury oprogramowania na wydajność i skalowalność systemu
- Znać pojęcie logiki biznesowej, znać najważniejsze zasady podziału systemów informatycznych na warstwy, znać różne rodzaje architektur w kontekście ewolucji historycznej metod konstruowania tego rodzaju oprogramowania
- Potrafić posługiwać się wzorcami warstwy prezentacji w procesie implementowania tej warstwy
- Potrafić posługiwać się wzorcami warstwy logiki biznesowej w procesie implementowania tej warstwy
- Potrafić używać wzorce integracyjne
- Być w stanie zrealizować komponenty transakcyjne systemu informatycznego
- Być w stanie zaimplementować bezpieczny mechanizm rozpraszania sesji w systemie biznesowym

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Martin Fowler, Architektura systemów zarządzania przedsiębiorstwem. Wzorce projektowe. (Tytuł oryginału: Patterns of Enterprise Application Architecture), Helion 2005
— Deepak Alur, John Crupi, Dan Malks, J2EE. Wzorce projektowe. Wydanie 2, Helion 2004

Literatura uzupełniająca przedmiotu:

— Gregor Hohpe, Bobby Woolf, Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions

Inne materiały dydaktyczne:

PROGRAMOWANIE NA PLATFORMIE SPRING FRAMEWORK

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) PROGRAMOWANIE NA PLATFORMIE SPRING FRAMEWORK			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy fakultatywny	
Rok: 3	**Semestr: 6**	**ECTS ogółem:4**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 2**	**studia niestacjonarne: 1**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Programowanie funkcyjne** **Programowanie współbieżne** **Wybrane środowiska programowania** **Języki i paradygmaty programowania**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: egzamin** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Konrad Grzanek**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	15	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	10
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	2
RAZEM:	**47**	**RAZEM:**	**22**
Praca własna studenta (PWS):	53	Praca własna studenta (PWS):	78
RAZEM z PWS:	**100**	**RAZEM z PWS:**	**100**

Sumaryczne obciążenie pracą studenta wg form aktywności:		
Forma aktywności:	**Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:**	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**47**	**22**
2. Przygotowanie się do zajęć	13	38
3. Przygotowanie esejów		
4. Wykonanie projektów	20	20
5. Zapoznanie z literaturą podstawową	20	20
6. Pisemna praca zaliczeniowa		
7. Inne:		
SUMA:	**100**	**100**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Spring Framework – przedstawienie technologii, jej elementów składowych oraz określenie znaczenia w ekosystemie technologicznym Java Enterprise Edition
2. Wzorzec IoC (Dependency Injection) i jego realizacja w ramie Spring – tworzenie ziaren, nazewnictwo i podstawowa konfiguracja
3. Wzorzec IoC – zaawansowane zagadnienia konfiguracyjne, planowanie architektury aplikacji klasy enterprise w oparciu o idiomy właściwe dla platformy. Zarządzanie zaawansowane cyklem życia ziaren. Zakresy ziaren
4. Adnotacje w języku Java – omówienie wprowadzające do ich użycia i tworzenia. IoC w Spring Frameworku, konfiguracja w oparciu o adnotacje
5. Walidacja danych, konwersje typów i formatowanie na platformie Spring.
6. Programowanie aspektowe – wprowadzenie. AOP na platformie Spring, prezentacja najważniejszych zagadnień. API Springa vs AspectJ.
7. Programowanie aspektowe, definiowanie aspektów, punktów przecięć i porad. Analiza wybranych przypadków użycia technologii.
8. Testowanie na platformie Spring: jednostkowe i integracyjne, używanie adnotacji.
9. Tworzenie algorytmów transakcyjnych, użycie deklaratywnego stylu zarządzania transakcjami. Programowe użycie transakcji
10. Użycie abstrakcji dla mechanizmów JDBC – wprowadzenie, najważniejsze zagadnienia architektoniczne
11. Operacje wsadowe JDBC, optymalizacja i upraszczanie operacji na bazie danych, modelowanie czynności bazodanowych w postaci obiektów języka Java, współpraca z bazami osadzonymi.
12. Przetwarzanie dokumentów XML, różne rodzaje marszalerów.
13. Framework MVC dla technologii WEB w ujęciu Spring, tworzenie kontrolerów, DispatcherServlet.Integracja z innymi realizacjami wzorca MVC: Struts, Tapestry.
14. Wybrane elementy warstwy integracji na platformie Spring: użycie technologii RMI, EJB oraz JMS
15. Uługa e-mail w oparciu o Spring Framework.

LABORATORIA:

Preferowaną formą zaliczenia laboratoriów jest zespołowa (2-3 osoby) realizacja całosemestralnego projektu polegającego na utworzeniu wybranej aplikacji biznesowej z wykorzystaniem frameworku Spring. Studenci powinni posłużyć się technologiami wskazanymi przez prowadzącego, te z kolei powinny odpowiadać treściom wykładowym.

Studenci dzielą się elementami funkcjonalnymi do wykonania a ich ocena końcowa zależy zarówno od indywidualnych osiągnięć w realizacji przydzielonych im zadań, jak i od całokształtu realizowanego zespołowo projektu.

Prowadzący zajęcia laboratoryjne powinien wspierać studentów swoimi uwagami i pomagać w rozwiązywaniu najtrudniejszych problemów. Powinien również dokonywać okresowej kontroli osiągnięć, tak całego zespołu, jak i poszczególnych jego członków.

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:

Wiedza:

Kod wg KRK:		**Kod KEK/ % udział przedmiotu w osiągnięciu efektu:**	**Metoda (forma) weryfikacji**
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Test wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych,inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego

Umiejętności:

Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego

Kompetencje społeczne:

Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań

* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?

Cele kształcenia (przedmiotowe efekty kształcenia):

Celem kształcenia jest zaznajomienie studentów z tworzeniem aplikacji biznesowych na podbudowie technologii Java w oparciu o ramę programową o nazwie Spring Framework.

Po zakończeniu kursu student:

- ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie tworzenia złożonych systemów biznesowych wykorzystujących technologię Spring Framework
- zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych służących celom biznesowym implementowanym w oparciu o programową ramę Spring
- ma umiejętność formułowania algorytmów i ich implementacji stosując Spring Framework; potrafi ocenić złożoność obliczeniową algorytmów realizujących logikę biznesową, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów stosując podejście do testowania jednostkowego oraz integracyjnego właściwą dla platformy Spring

W szczególności student powinien:

- znać architekturę ramy programowej Spring, rozumieć jej znaczenie w kontekście użycia do budowy złożonych systemów biznesowych,
- potrafić posługiwać się mechanizmem (wzorcem projektowym) IoC (Dependency Injection) w realizacji Spring w celu zapewnienia współpracy pomiędzy ziarnami biznesowymi,
- znać podstawowe zagadnienia programowania aspektowego, potrafić je wykorzystywać w praktyce w oparciu o mechanizmy dostarczane przez ramę programową Spring,
- znać zasady tworzenia testów jednostkowych i integracyjnych w oparciu o mechanizmy dostarczane przez Spring,
- umieć stosować prawidłowo transakcyjność, realizować mechanizmy transakcyjne z wykorzystaniem API Springa,
- realizować algorytmy biznesowe wymagające współpracy z relazyjnymi i nierelacyjnymi bazami danych poprzez API JDBC z wykorzystaniem poznanych klas użyteczności należących do Spring Frameworku,
- znać i stosować mechanizmy przetwarzania danych XML na platformie Spring
- potrafić implementować widoki WEB dla aplikacji tworzonych w technologii Spring, znać najważniejsze elementy architektury implementacji wzorca MVC
- stosować różne elementy integracyjne właściwe dla Springa

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Clarence Ho, Rob Harrop, Pro Spring 3, Apress; 1 edition (2012), ISBN-10: 1430241071, ISBN-13: 978-1430241072
— Craig Walls, Spring in Action, Manning Publications; Third Edition edition (2011), ISBN-10: 1935182358, ISBN-13: 978-1935182351

Literatura uzupełniająca przedmiotu:

— Gary Mak, Daniel Rubio, Josh Long, Spring Recipes: A Problem-Solution Approach, Apress; 2nd edition (2010), ISBN-10: 1430224991, ISBN-13: 978-1430224990

Inne materiały dydaktyczne:

— Spring Framework Reference Documentation 3.2.0.RELEASE,
— http://static.springsource.org/spring/docs/3.2.x/spring-framework-reference/pdf/spring-framework-reference.pdf
— Spring Framework API, http://static.springsource.org/spring/docs/3.2.x/javadoc-api/

WARSTWA WIDOKU W UJĘCIU RAMOWYM

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy fakultatywny	
Rok: 3	**Semestr: 6**	**ECTS ogółem:4**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 2**	**studia niestacjonarne: 1**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Podstawy programowania,** **Wybrane środowiska programowania** **Języki i paradygmaty programowania**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi.** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: egzamin ustny** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Zbigniew Filutowicz**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	15	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	10
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	2
RAZEM:	**47**	**RAZEM:**	**22**
Praca własna studenta (PWS):	53	Praca własna studenta (PWS):	78
RAZEM z PWS:	**100**	**RAZEM z PWS:**	**100**
Sumaryczne obciążenie pracą studenta wg form aktywności:			
Forma aktywności:		**Szacowana liczba godzin potrzebnych na zrealizowanie**	

	aktywności:	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**47**	**22**
2. Przygotowanie się do zajęć	23	43
3. Przygotowanie esejów	5	5
4. Wykonanie projektów	10	10
5. Zapoznanie z literaturą podstawową	10	15
6. Pisemna praca zaliczeniowa	5	5
7. Inne:		
SUMA:	**100**	**100**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Pojęcia biblioteki w programowaniu, software framework (platforma programistyczna, struktura, szkielet, rama), aplikacja, program, application framework, wzorce projektowe, zasda Don't repeat yourself - wielokrotne użycie kodu, maszyna wirtualna.
2. Inżynieria oprogramowania, paradygmaty programowania, programowanie OOP. Programowanie komponentowe, Web Services, Component-based software engineering (CBSE), component-based development (CBD), modular programming
3. VPL Visual programming language, Środowiska programistyczne CASE. Kreatory (generatory) aplikacji.
4. Web application framework. RIA. Application framework, software framework.
5. Adobe AIR, Ajax i swf
6. MS ASP.NET, .NET Framework, Silverlight
7. OpenLaszlo
8. JavaFX i FXML , JavaServer Face.
9. Google Web Toolkit
10. Dhtmlx,
11. Standardy i rozwój platform programistycznych bogatych aplikacji webowych.

LABORATORIA

1. Organizacja zajęć. Omówienie zasad wykonania projektów oraz zasad zaliczenia zajęć. Wprowadzenie w tematykę przedmiotu. Omówienie zasobów wiedzy bibliograficznej i netograficznej.
2. Analiza przykładowych aplikacji typu RIA.
3. Przykładowy projekt bogatej aplikacji webowej wykonywany w dwóch wybranych technologiach implementacyjnych (Web application framework)
4. Wspólne referowanie opracowanych projektów i dyskusja uzyskanych wyników oraz wnioski

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:

Wiedza:

Kod wg KRK:		**Kod KEK/%: udział przedmiotu w osiągnięciu efektu:**	**Metoda (forma) weryfikacji***

T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Egzamin, ustny sprawdzian wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Egzamin, ustny sprawdzian wiedzy
Umiejętności:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	Ocena zadań projektowych oraz obserwacja wykonania zadań praktycznych
Kompetencje społeczne:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań

* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?

Cele kształcenia (przedmiotowe efekty kształcenia):

Celem zajęć jest przedstawienie podstawowych zagadnień i wyrobienie umiejętności z zakresu inżynierii oprogramowania opartej na platformach programistycznych (framework).

Na zajęciach laboratoryjnych studenci analizują przykładowe wykorzystania platform programistycznych - frameworków oraz zdobywają umiejętności ich stosowania ich w wybranych projektach informatycznych. Studenci implementują opracowane projekty interfejsów użytkownika z wykorzystaniem platform programistycznych, pod nadzorem prowadzącego.

Po ukończeniu kursu student:

- Ma wiedzę z zakresy inżynierii oprogramowania opartej na platformach programistycznych (framework).
- Poznaje wybrane platformy programistyczne (framework) bogatych aplikacji webowych stosowanych w inżynierii oprogramowania i rozumie zasady ich używania.
- Potrafi ocenić wybrane platformy programistyczne (framework) bogatych aplikacji webowych .
- Potrafi ocenić przydatność platform programistycznych (framework) bogatych aplikacji webowych w konkretnych rozwiązaniach.
- Potrafi używać wybrane platformy programistyczne (framework) do programowania systemów oprogramowania.
- Potrafi wykorzystywać oprogramowanie do komputerowego wspomagania projektowania i

implementacji projektów informatycznych z wykorzystaniem środowisk programistycznych IDE.

- Ma umiejętność przekazywania wiedzy i dyskusji na profesjonalnym w zakresie platform programistycznych (framework) bogatych aplikacji webowych

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Mike Snell, Lars Powers, Microsoft Visual Studio 2010. Księga eksperta, Helion 2010
— Adam Tacy, Robert Hanson, Jason Essington and Anna Tokke, GWT in Action, Amazon 2012
— Google Web Toolkit, https://developers.google.com/web-toolkit/gettingstarted?hl=pl
— JavaFX http://www.oracle.com/technetwork/java/javafx/overview/index.html

Literatura uzupełniająca przedmiotu:

— Larry Ullman, Adobe Air i Ajax. Szybki start, Helion 2010
— Earn more on Android, development http://www.java2s.com/Code/Java/GWT/CatalogGWT.htm

Inne materiały dydaktyczne:

— Lista RIA frameworks http://en.wikipedia.org/wiki/List_of_rich_Internet_application_frameworks
— Lista bibliotek JavaScript http://en.wikipedia.org/wiki/List_of_JavaScript_libraries
— Web application framework http://en.wikipedia.org/wiki/Web_application_framework
— http://www.openlaszlo.org
— http://www.microsoft.com/expression/
— http://silverlight.net/quickstarts/
— http://labs.adobe.com/technologies/flex/
— http://www.tryadobeair.com/
— http://www.sun.com/software/javafx/index.jsp

SYSTEMY MASOWEGO PRZETWARZANIA INFORMACJI (HADOOP)

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) SYSTEMY MASOWEGO PRZETWARZANIA INFORMACJI (HADOOP)			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy fakultatywny	
Rok: 4	**Semestr: 7**	**ECTS ogółem:5**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 3**	**studia niestacjonarne: 2**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Bazy danych i aplikacje** **Wybrane środowiska programowania** **Języki i paradygmaty programowania** **Programowanie funkcyjne**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: egzamin** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Konrad Grzanek**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	30	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	20
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	3
RAZEM:	**62**	**RAZEM:**	**33**
Praca własna studenta (PWS):	63	Praca własna studenta (PWS):	92

RAZEM z PWS:	125	RAZEM z PWS:	125

Sumaryczne obciążenie pracą studenta wg form aktywności:

Forma aktywności:	Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**62**	**33**
2. Przygotowanie się do zajęć	13	22
3. Przygotowanie esejów		
4. Wykonanie projektów	20	40
5. Zapoznanie z literaturą podstawową	30	30
6. Pisemna praca zaliczeniowa		
7. Inne:		
SUMA:	**125**	**125**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Wprowadzenie do zagadnień masowego składowania i przetwarzania informacji, algorytm MapReduce firmy Google, Google File System (GFS), motywacje, ograniczenia. Historia Hadoop, najważniejsze elementy architektury.
2. Instalacja i konfiguracja klastra Hadoop, najważniejsze zagadnienia administracyjne; analiza logów, narzędzia, rutynowe działania administratorskie.
3. HDFS (Hadoop Distributed File System) – przedstawienie cech, węzły nazw i danych, interfejs command-line, interfejs dla języka Java, anatomia zapisu i odczytu.
4. Operacje wejścia-wyjścia, zachowywanie integralności danych, kompresja, serializacja, plikowe struktury danych.
5. Tworzenie algorytmów wykorzystujących wzorzec MapReduce, konfiguracja i uruchamianie w klastrze.
6. Anatomia procesu realizującego algorytm MapReduce, zadania, wystąpienia błędów, aspekty wydajnościowe.
7. Formaty wejściowe i wyjściowe mechanizmów MapReduce, typy danych występujące w realizacji algorytmów opartych o wzorzec.
8. Cechy funkcjonalne MapReduce; zliczanie, sortowanie, złączenia, klasy biblioteczne.
9. Język Pig, omówienie, instalacja i użycie z Hadoop, analiza przykładowych algorytmów.
10. Pig – budowa kompleksowego rozwiązania algorytmicznego dla wybranego obszaru zastosowań.
11. System BigTable firmy Google – omówienie. System HBase w Hadoop jako odpowiednik BigTable. Instalacja, konfiguracja, użycie.
12. HBase – analiza wybranej aplikacji, studium przypadku i przedstawienie zaawansowanych cech.
13. Testowanie algorytmów w środowisku Hadoop/HBase/Pig.
14. Planowanie rozwiązań pozwalających na zarządzanie wielkimi zbiorami danych w oparciu o przedstawioną technologię. Aspekty biznesowe, kosztorysowanie, zagadnienia sprzętowe i związane z zarządzaniem energią.
15. Przedstawienie interfejsu do Hadoop dla języka Clojure – Cascalog. Podsumowanie wykładu.

LABORATORIA:

Prefererowaną formą zaliczenia laboratoriów jest zespołowa (2-3 osoby) realizacja całosemestralnego projektu polegającego na utworzeniu systemu zarządzającego rozległymi zbiorami danych oraz pozwalającego na ich przetwarzanie z wykorzystaniem technologii HDFS/Hadoop/Pig/HBase.

Studenci dzielą się elementami funkcjonalnymi do wykonania a ich ocena końcowa zależy zarówno od indywidualnych osiągnięć w realizacji przydzielonych im zadań, jak i od całokształtu realizowanego zespołowo projektu.

Projekt powinien być realizowany w pracowni komputerowej. Sugeruje się użycie 3-węzłowej (domyślnej) architektury klastra dla jednego projektu. Proponowane etapy projektu:

- Wybór tematyki do opracowania – problemu związanego z koniecznością gromadzenia i przetwarzania dużych zbiorów danych.
- Instalacja Hadoop i konfiguracja klastra.
- Wprowadzenie danych do systemu (w tym – generacja przykładowych danych mechanizmem programowym symulującym przyrostowe ich powstawanie).
- Budowa mechanizmów analitycznych z wykorzystaniem MapReduce w języku Java.
- Budowa mechanizmów analitycznych z wykorzystaniem języka Pig.
- (PUNKT OPCJONALNY) Wykorzystanie Cascalog do tworzenia mechanizmów analitycznych.

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW KSZTAŁCENIA, CELE KSZTAŁCENIA

Efekty kształcenia:

Wiedza:

Kod wg KRK:		**Kod KEK/ % udział przedmiotu w osiągnięciu efektu:**	**Metoda (forma) weryfikacji**
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Test wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych,inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego

Umiejętności:

Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	Test wiedzy + obserwacja i ocena wykonania zadania praktycznego

Kompetencje społeczne:

Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej	K_K02/1%	obserwacja studenta i ocena wykonania

InzA_K01	działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje		zadań

* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?

Cele kształcenia (przedmiotowe efekty kształcenia):

W trakcie kursu studenci poznają technologię Hadoop pozwalającą na utworzenie klastra zarządzającego wielkimi zbiorami danych. Uczą się realizować algorytmy analizujące dane dla tej technologii tworzone z wykorzystaniem wzorca MapReduce. Po ukończeniu kursu student:

- ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie zarządzania rozległymi zbiorami danych oraz ich analizowania, implementacji mechanizmów składowania i przetwarzania informacji w systemie Hadoop
- zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych zarządzających i przetwarzających rozległe zbiory danych w oparciu o HDFS/Hadoop/Pig
- ma umiejętność formułowania algorytmów analizujących rozległe dane oraz ich implementacji stosując technologie HDFS/Hadoop/Pig, oraz (opcjonalnie) Cascalog

W szczególności student powinien:

- znać zasadę działania mechanizmu MapReduce, rozumieć zasady działania Google File System, znać uwarunkowania związane z utrzymaniem systemów informatycznych, w których powstają ogromne zbiory danych o kluczowym dla przedsięwzięcia znaczeniu
- umieć budować klastry Hadoop i administrować nimi (w podstawowym zakresie)
- znać najważniejsze cechy rozproszonego systemu plików HDFS, rozumieć zasady jego działania
- być w stanie umieścić w klastrze dane wygenerowane programowo, poprzez interfejs w języku Java
- umieć budować algorytmy analityczne działające na składowanych danych zgodnie ze schematem MapReduce w języku Java
- znać język Pig, potrafić używać go do stworzenia algorytmów analitycznych
- rozumieć zasadę działania, znać zastosowania systemu HBase, potrafić posługiwać się nim
- potrafić tworzyć testy na platformie Hadoop
- umieć zaplanować i zaimplementować przedsięwzięcie polegające na zarządzaniu i analizowaniu rozległych zbiorów danych w oparciu o technologię Hadoop oraz technologie stowarzyszone z nią

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Tom White, Hadoop: The Definitive Guide, O'Reilly Media; Third Edition edition (2012), ISBN-10: 1449311520, ISBN-13: 978-1449311520
— Chuck Lam, Hadoop in Action, Manning Publications; 1st edition (2010), ISBN-10: 1935182196, ISBN-13: 978-1935182191

Literatura uzupełniająca przedmiotu:

— Eric Sammer, Hadoop Operations, O'Reilly Media; 1st edition (2012), ISBN-10: 1449327052, ISBN-13: 978-1449327057
— Donald Miner, Adam Shook, MapReduce Design Patterns: Building Effective Algorithms and Analytics for Hadoop and Other Systems, O'Reilly Media (2012), ISBN-10: 1449327176, ISBN-13: 978-1449327170
— Henry H. Liu, Hadoop Essentials: A Quantitative Approach, CreateSpace Independent Publishing Platform (2012), ISBN-10: 1480216372, ISBN-13: 978-1480216372

Inne materiały dydaktyczne:

— Hadoop API Documentation: http://hadoop.apache.org/docs/current/api/index.html
— HDFS User Guide: http://hadoop.apache.org/docs/r0.18.2/hdfs_user_guide.html

WARSTWY INTEGRACJI W WYBRANYCH ŚRODOWISKACH

I. OGÓLNE INFORMACJE PODSTAWOWE O PRZEDMIOCIE (MODULE)			
Nazwa przedmiotu (modułu) WARSTWY INTEGRACJI W WYBRANYCH ŚRODOWISKACH			
Nazwa jednostki organizacyjnej prowadzącej kierunek:		**Wydział Studiów Międzynarodowych i Informatyki Społecznej Akademii Nauk w Łodzi**	
Nazwa kierunku studiów i poziom kształcenia:		INFORMATYKA studia I stopnia	
Nazwa specjalności:		Technologie programowania	
Język wykładowy: polski	**Rodzaj modułu kształcenia:**	specjalnościowy fakultatywny	
Rok: 4	**Semestr: 7**	**ECTS ogółem:5**	**Data aktualizacji sylabusa: 2012.10.01**
ECTS (zajęcia z bezpośrednim udziałem nauczyciela akademickiego i studenta):		**studia stacjonarne: 3**	**studia niestacjonarne: 2**
ECTS (zajęcia praktyczne):		**studia stacjonarne: 1**	**studia niestacjonarne: 1**
Wymagania wstępne w zakresie wiedzy oraz umiejętności:		**Podstawy programowania,** **Wybrane środowiska programowania** **Języki i paradygmaty programowania**	
Forma prowadzenia zajęć i metody dydaktyczne:		**Wykład prowadzony metodą podającą wspomagany prezentacjami multimedialnymi.** **Laboratorium prowadzone w pracowni komputerowej**	
Forma zaliczania przedmiotu:		**Wykład: egzamin ustny** **Laboratorium: zaliczenie**	
Katedra (Zakład) odpowiedzialna za przedmiot:		**Instytut Technologii Informatycznych**	
Osoba koordynująca przedmiot:		**dr inż. Zbigniew Filutowicz**	

II. WYMIAR GODZINOWY ZAJĘĆ ORAZ INDYWIDUALNEJ PRACY WŁASNEJ STUDENTA			
Ogólna liczba godzin zajęć dydaktycznych na studiach stacjonarnych i niestacjonarnych z podziałem na formy:			
S t u d i a s t a c j o n a r n e		**S t u d i a n i e s t a c j o n a r n e**	
Wykład:	30	Wykład:	10
Ćwiczenia:		Ćwiczenia:	
Laboratorium:	30	Laboratorium:	20
Ćwiczenia projektowe:		Ćwiczenia projektowe:	
Warsztaty:		Warsztaty:	
Seminarium:		Seminarium:	
Zajęcia terenowe:		Zajęcia terenowe:	
Praktyki:		Praktyki:	
E/Z	2	E/Z	3
RAZEM:	**62**	**RAZEM:**	**33**
Praca własna studenta (PWS):	63	Praca własna studenta (PWS):	92
RAZEM z PWS:	**125**	**RAZEM z PWS:**	**125**
Sumaryczne obciążenie pracą studenta wg form aktywności:			

Forma aktywności:	Szacowana liczba godzin potrzebnych na zrealizowanie aktywności:	
	studia stacjonarne	**studia niestacjonarne**
1. Godziny realizowane w bezpośrednim kontakcie z nauczycielem akademickim	**62**	**33**
2. Przygotowanie się do zajęć	33	53
3. Przygotowanie esejów	5	5
4. Wykonanie projektów	10	10
5. Zapoznanie z literaturą podstawową	10	19
6. Pisemna praca zaliczeniowa	5	5
7. Inne:		
SUMA:	**125**	**125**

III. TREŚCI KSZTAŁCENIA

Treści kształcenia (uszczegółowione, zaprezentowane z podziałem na poszczególne formy zajęć):

WYKŁADY:

1. Pojęcia wzorca architektonicznego (Architectural pat tern) w inżynierii oprogramowania.
2. Systemy rozproszone i ich integracja.
3. Warstwa integracji logiki biznesowej i baz danych.
4. Inżynieria oprogramowania. Architektura zorientowana na usługi SOA, Web-oriented architekture , Web Services. WSDL, SOAP.
5. Web 2.0 i Web 3.0, witryny mashupowe.
6. Software agents, Systemy agentowe. Agent-based system Agent-based model, multi-agent system, Inżynieria agentowa, paradygmaty programowania,
7. Platformy programistyczne - framework do programowania agentowego.
8. Agenci interfejsu użytkownika.
9. Workflow, Enterprise architecture Framework.
10. Systemy zintegrowane. Systemy przemysłowe, gospodarcze i zarządzanie. Oprogramowanie aplikacyjne i komputerowe wspomaganie technologii CAx.
11. P2P Peer-to-Peer
12. Cloud Computing, private Cloud Computing. Wspomaganie informatyczne własnego przedsięwzięcia gospodarczego w oparciu o Cloud Computing i oprogramowanie aplikacyjne (użytkowe)
13. Platformy implementacji chmury, Windows Azure, Amazon Web Services, Google App Engine i inne
14. Software as a Services, SaaS, infrastructure as a service (IaaS), platform as a service (PaaS), desktop as a service (DaaS), and backend as a service (BaaS)
15. Standardy i rozwój platform programistycznych dla aplikacji rozproszonych i ich integracji.

LABORATORIA:

1. Organizacja zajęć. Omówienie zasad wykonania projektów oraz zasad zaliczenia zajęć. Wprowadzenie w tematykę przedmiotu. Omówienie zasobów wiedzy bibliograficznej i netograficznej.
2. Przykładowy projekt aplikacji opartej na Web Services integrującej różne środowiska (platformy implementacyjne).
3. Badanie wybranego środowiska implementacyjnego do obliczeń w Cloud Computing (Microsoft Windows Azure lub Google App Engine) oraz projekt własnej aplikacji dla tego środowiska.
4. Wspólne referowanie opracowanych projektów i dyskusja uzyskanych wyników oraz wnioski

IV. EFEKTY KSZTAŁCENIA (OBSZAROWE I KIERUNKOWE) WRAZ Z WERYFIKACJĄ EFEKTÓW

KSZTAŁCENIA, CELE KSZTAŁCENIA			
Efekty kształcenia:			
Wiedza:			
Kod wg KRK:		**Kod KEK/%: udział przedmiotu w osiągnięciu efektu:**	**Metoda (forma) weryfikacji***
T1A_W03 T1A_W04 T1A_W05	ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie algorytmów i ich złożoności, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i technologii multimedialnych, komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W05/2%	Egzamin, ustny sprawdzian wiedzy
T1A_W04 T1A_W05 T1A_W07 InzA_W02 InzA_W05	zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu nieskomplikowanych zadań informatycznych z zakresu analizy, projektowania i budowy systemów informatycznych, systemów operacyjnych, sieci komputerowych i systemów rozproszonych, grafiki i systemów multimedialnych, sztucznej inteligencji, baz danych, inżynierii oprogramowania oraz bezpieczeństwa systemów informatycznych	K_W07/2%	Egzamin, ustny sprawdzian wiedzy
Umiejętności:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_U09 T1A_U13 T1A_U15 InzA_U02 InzA_U05 InzA_U07	ma umiejętność formułowania algorytmów i ich implementacji stosując przynajmniej jedno z powszechnie używanych środowisk programistycznych; potrafi ocenić złożoność obliczeniową algorytmów, optymalizować je, odszukać w nich słabości i błędy oraz opracować plan testów	K_U08/5%	Ocena zadań projektowych oraz obserwacja wykonania zadań praktycznych
Kompetencje społeczne:			
Kod wg KRK:		**Kod KEK/%:**	**Metoda (forma) weryfikacji**
T1A_K02 T1A_K04 InzA_K01	ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka, w tym wpływ tej działalności na środowisko i związaną z tym odpowiedzialność za podejmowane decyzje	K_K02/1%	obserwacja studenta i ocena wykonania zadań
* test wiedzy, ustny sprawdzian wiedzy, praca pisemna, praca pisemna z obroną, prezentacja, zadanie praktyczne lub projektowe, zadanie zespołowe z indywidualną kontrolą osiągnięć, obserwacja i ocena wykonania zadania praktycznego, kontrola i ocena przebiegu praktyk, inna – jaka?			

Cele kształcenia (przedmiotowe efekty kształcenia):

Celem zajęć jest przedstawienie podstawowych zagadnień i wyrobienie umiejętności z zakresu inżynierii oprogramowania dotyczącej obliczeń rozproszonych , ich integracji i Cloud Computing.

Na zajęciach laboratoryjnych studenci analizują przykładowe wykorzystania Web Services i private Cloud Computing oraz zdobywają umiejętności ich stosowania ich w wybranych projektach informatycznych. Studenci implementują opracowane algorytmy przetwarzania danych pod kątem integracji różnych platform, pod nadzorem prowadzącego.

Po ukończeniu kursu student:

- Ma wiedzę z zakresy inżynierii oprogramowania systemów rozproszonych i ich integracji.
- Poznaje wybrane platformy programistyczne w inżynierii oprogramowania i rozumie zasady ich

używania.

- Poznaje wybrane platformy implementacyjne do obliczeń Cloud Computing.
- Potrafi ocenić przydatność platform programistycznych dla konkretnych zastosowań.
- Potrafi używać platform programistycznych do implementowania konkretnych systemów oprogramowania.
- Potrafi wykorzystywać oprogramowanie do komputerowego wspomagania projektowania i implementacji projektów informatycznych z wykorzystaniem środowisk programistycznych IDE.
- Ma umiejętność przekazywania wiedzy i dyskusji na profesjonalnym w zakresie inżynierii wzorców projektowych.

V. LITERATURA PRZEDMIOTU ORAZ INNE MATERIAŁY DYDAKTYCZNE

Literatura podstawowa przedmiotu:

— Adriaan de Jonge, Google App Engine. Tworzenie wydajnych aplikacji w Javie, Helion 2012
— Tejaswi Redkar, Tony Guidici, Platforma Windows Azure, Helion 2012
— Szyperski C., Oprogramowanie komponentowe, obiekty za mało, inżynieria oprogramowania, WNT 1998.

Literatura uzupełniająca przedmiotu:

— Arthur Mateos, Jothy Rosenberg, ‚Chmura obliczeniowa. Rozwiązania dla biznesu, Helion 2011
— Amy Shuen, Web 2.0. Przewodnik po strategiach, Helion 2009
— Dan Sanderson, Programming Google App Engine, 2nd Edition, Build & Run Scalable Web Applications on Google's Infrastructure, Publisher: O'Reilly Media, Released: October 2012

Inne materiały dydaktyczne:

— Component-based software engineering http://en.wikipedia.org/wiki/Category:Component-based_software_engineering
— Programowanie agentowe http://www.codeguru.pl/baza-wiedzy/programowanie-agentowe-agent-behawioralny,2492