KARTA OPISU MODUŁU KSZTAŁCENIA

Nazwa modułu/przedmiotu	Kod
Bezpieczeństwo systemów informatycznych	**1010514371010500599**

Kierunek studiów	Profil kształcenia (ogólnoakademicki, praktyczny)	Rok / Semestr
Informatyka	**ogólnoakademicki**	**4 / 7**
Ścieżka obieralności/specjalność	Przedmiot oferowany w języku:	Kurs (obligatoryjny/obieralny)
-	**polski**	**obligatoryjny**

Stopień studiów:	**Forma studiów** (stacjonarna/niestacjonarna)
I stopień	**niestacjonarna**

Godziny	Liczba punktów
Wykłady: **14** Ćwiczenia: **-** Laboratoria: **16** Projekty/seminaria: **-**	**4**

Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny)	(ogólnouczelniany, z innego kierunku)
kierunkowy	**z danego kierunku**

Obszar(y) kształcenia i dziedzina(y) nauki i sztuki	**Podział ECTS (liczba i %)**

Odpowiedzialny za przedmiot / wykładowca:

dr inż. Michał Szychowiak
email: Michal.Szychowiak@cs.put.poznan.pl
tel. 61 665 2964
Wydział Informatyki
ul. Piotrowo 2 60-965 Poznań

Odpowiedzialny za przedmiot / wykładowca:

mgr inż. Bartosz Brodecki
email: Bartosz.Brodecki@cs.put.poznan.pl
tel. 61 665 2952
Wydział Informatyki
ul. Piotrowo 2, 60-965 Poznań

Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:

1	**Wiedza:**	Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę w zakresie algorytmów: oceny ich poprawności i złożoności (czasowej, złożoności średniej oraz w najgorszym przypadku), znać podstawowe zasady programowania strukturalnego i/lub obiektowego. Powinien również znać podstawowe metody, techniki i narzędzia pomocne w budowie programów. Powinien posiadać także podstawową wiedzę na temat zagadnień związanych z programowaniem wielowątkowym, w szczególności znać problemy związane z wzajemnym wykluczaniem oraz zakleszczaniem procesów i wątków.
2	**Umiejętności:**	Powinien sprawnie posługiwać się możliwościami udostępnianymi przez system operacyjny, przy czym preferowany jest system UNIX/Linux. Powinien również posiadać umiejętność formułowania algorytmów i ich programowania z użyciem języka C lub C++. Powinien posiadać umiejętność rozwiązywania podstawowych problemów związanych z programowaniem w systemie sekwencyjnym, oraz umiejętność pozyskiwania informacji ze wskazanych źródeł, w tym także ze źródeł w języku angielskim.
3	**Kompetencje społeczne**	Powinien również rozumieć konieczność poszerzania swoich kompetencji oraz mieć gotowość do podjęcia współpracy w ramach zespołu. Ponadto w zakresie kompetencji społecznych student musi prezentować takie postawy jak uczciwość, odpowiedzialność, wytrwałość, ciekawość poznawcza, kreatywność, kultura osobista, szacunek dla innych ludzi.

Cel przedmiotu:

1. Przekazanie studentom podstawowej wiedzy na temat specyfiki systemów rozproszonych i podstawowych różnic dzielących ich od systemów ściśle powiązanych, budowy systemów rozproszonych, analizy złożoności komunikacyjnej i czasowej algorytmów przy uwzględnieniu specyfiki systemów rozproszonych, analizy poprawności algorytmów rozproszonych
2. Zaznajomienie studentów z trendami rozwoju systemów rozproszonych, podstawowymi problemami pojawiającymi się w trakcie tworzenia systemów rozproszonych i realizacji ich typowych zadań, oraz ich rozwiązaniami
3. Umożliwienie studentom nabycia umiejętności konstrukcji aplikacji rozproszonych, posługiwania się wybranymi narzędziami służącymi do implementacji programów w rozproszonym środowisku
4. Rozwijanie u studentów umiejętności rozwiązywania prostych problemów wymagających rozwiązań specyficznych dla systemów rozproszonych, takich jak wyznaczanie globalnego stanu spójnego
5. Wspomaganie kształtowania u studentów umiejętności pracy zespołowej, właściwych nawyków programistycznych, takich jak dokumentowanie kodu.
6. Kształtowanie umiejętności optymalizacji programów, poprzez dobór odpowiednich narzędzi, algorytmów i metod implementacji

Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia

Wiedza:

1. ma uporządkowaną, podbudowaną teoretycznie wiedzę ogólną w zakresie algorytmów przetwarzania rozproszonego i ich złożoności - [K_W4]

2. ma szczegółową wiedzę nt. algorytmiki przetwarzania rozproszonego - [K_W5]

3. ma wiedzę o trendach rozwojowych i najistotniejszych nowych osiągnięciach w informatyce, w szczególności odnośnie systemów rozproszonych - [K_W6]

4. zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu prostych zadań informatycznych z zakresu budowy rozproszonych systemów komputerowych, - [K_W8]

5. ma uporządkowaną, podbudowaną teoretycznie podstawową wiedzę w zakresie architektury systemów rozproszonych - [K_W4]

6. zna podstawowe zadania realizowane przez aplikacje rozproszone i typowe problemy pojawiające się w trakcie realizacji tych zadań - [-]

7. ma wiedzę niezbędną do identyfikacji typowych problemów oraz wyboru właściwych rozwiązań - [-]

8. ma podstawową wiedzę na temat zagadnień związanych z niezawodnością systemów rozproszonych - [-]

Umiejętności:

1. potrafi ocenić złożoność obliczeniową algorytmów i problemów z zakresu przetwarzania rozproszonego - [K_U13]

2. potrafi ocenić architekturę oprogramowania z punktu widzenia wymagań pozafunkcjonalnych - [K_U15]

3. potrafi - zgodnie z zadaną specyfikacją - zaprojektować oraz zrealizować prosty rozproszony system informatyczny, używając właściwych metod, technik i narzędzi - [K_U21]

4. ma umiejętność formułowania algorytmów rozproszonych i ich programowania z użyciem bibliotek implementujących standardy MPI oraz PVM - [K_U22]

5. potrafi przygotować, w języku ojczystym i angielskim, dobrze udokumentowane opracowanie problemów z zakresu systemów rozproszonych - [K_U3]

6. potrafi wybrać narzędzie programistyczne odpowiednie do danego zadania programistycznego - [K_U20]

7. potrafi opisać specyficzne cechy systemów rozproszonych - [-]

8. potrafi porównać opisać i zdefiniować podstawowe, typowe problemy związane z realizacją typowych zadań systemów rozproszonych, takich jak niezawodna komunikacja, wyznaczanie konsensusu, wyznaczanie spójnego stanu globalnego - [-]

9. potrafi zdefiniować podstawowe pojęcia związane z systemami rozproszonymi - [-]

10. potrafi zidentyfikować problemy pojawiające się przy budowie aplikacji rozproszonej oraz zastosować znane sobie, typowe rozwiązania tego problemu - [-]

11. potrafi przeanalizować algorytm oraz jego implementację w celu wskazania źródeł potencjalnych problemów z efektywnością oraz poprawnością - [-]

Kompetencje społeczne:

1. rozumie, że w informatyce wiedza i umiejętności bardzo szybko stają się przestarzałe - [K_K1]

2. zna przykłady i rozumie przyczyny wadliwie działających rozproszonych systemów informatycznych, które doprowadziły do poważnych strat finansowych, społecznych - [K_K4]

3. potrafi współdziałać i pracować w grupie, przyjmując w niej różne role - [K_K5]

Sposoby sprawdzenia efektów kształcenia

Efekty kształcenia przedstawione wyżej weryfikowane są w następujący sposób:

Ocena formująca:

a) w zakresie wykładów:

- na podstawie odpowiedzi na pytania dotyczące materiału omówionego na poprzednich wykładach;

b) w zakresie ćwiczeń:

- na podstawie oceny bieżącego postępu realizacji zadań,

Ocena podsumowująca:

Sprawdzanie założonych efektów kształcenia realizowane jest przez:

- ocenę umiejętności związanych z realizacją ćwiczeń laboratoryjnych,
- ocenę sprawozdania przygotowywanego częściowo w trakcie zajęć, a częściowo po ich zakończeniu; ocenę przygotowanego przez zespoły studentów projektu w wybranym przez niego języku programowania (spośród C, C++ lub Java) wykorzystującym MPI, PVM lub inne środowisko, przy czym studenci deklarują wkład każdej osoby w realizację projektu
- ocenę i ?obronę? przez studenta sprawozdania z realizacji projektu, ocena ta obejmuje umiejętność pracy w zespole, uzasadnienie wyboru rozwiązania w zależności od jego złożoności i przydatności w określonej architekturze systemu, oraz analizę poprawności i złożoności czasowej i komunikacyjnej rozwiązania.
- ocenę wiedzy i umiejętności wykazanych na kolokwium zaliczeniowym o charakterze problemowym, obejmującego 4 pytania sprawdzające zagadnienia szczegółowo omawiane w trakcie wykładów. Wymagane jest zdobycie co najmniej 7 punktów na 12 możliwych.

Uzyskiwanie punktów dodatkowych za aktywność podczas zajęć, a szczególnie za:

- omówienia dodatkowych aspektów zagadnienia,
- efektywność zastosowania zdobytej wiedzy podczas rozwiązywania zadanego problemu,
- umiejętność współpracy w ramach zespołu praktycznie realizującego zadanie szczegółowe w laboratorium,
- uwagi związane z udoskonaleniem materiałów dydaktycznych,
- wskazywanie trudności percepcyjnych studentów umożliwiające bieżące doskonalenia procesu dydaktycznego.
- Aktywne uczestnictwo w czasie dyskusji mających na celu rozwiązanie postawionych problemów oraz ich rozwiązania

Treści programowe

W ramach wykładu student zapoznaje się z rzeczywistymi przykładami istniejących systemów rozproszonych, poznaje najważniejsze cechy decydujące o specyfice systemów tego rodzaju, poznaje powody tworzenia tego rodzaju systemów.

W dalszej kolejności przedstawiane są podstawowe pojęcia i definicje związane z tematyką przetwarzania rozproszonego: proces rozproszony i sekwencyjny. Student poznaje model formalny procesu sekwencyjnego i procesu rozproszonego oraz pojęcia związane z wykonaniem procesu i historią wykonania. Poznaje zagadnienia związane z aktywnością procesu, o warunkach uaktywnienia, klasycznych modelach żądań. Dalej przedstawione są pojęcia i formalne definicje kanałów komunikacyjnych, predykatów opisujących stan kanału, operacji komunikacyjnych. Przedstawione są także różnice między komunikacją synchroniczną i asynchroniczną. Wykład przedstawia także różne topologie przetwarzania rozproszonego, właściwości przetwarzania rozproszonego (relacja poprzedzania, diagramy przestrzenno-czasowe, grafy stanów osiągalnych, niedeterminizm przetwarzania,

Następnie student poznaje zagadnienia związane z realizacją czasu logicznego (wirtualnego), oraz poznaje algorytmy Matterna i Lamporta realizujące mechanizmy zegarów wektorowych i skalarnych. Omawiane są także zagadnienia związane z warunkami poprawności algorytmów rozproszonych oraz z analizą złożoności czasowej i komunikacyjnej algorytmów rozproszonych.

Kolejnym zagadnieniem omawianym na wykładzie jest fundamentalne pojęcie spójnego stanu globalnego. Po wprowadzeniu podstawowych definicji (konfiguracji, konfiguracji spójnej, odcięcia i odcięcia spójnego, linii odcięcia) pokazane są możliwe zastosowania algorytmów wyznaczania stanów globalnych (m.in. przedstawione jest pojęcie predykatów globalnych). Wyjaśnione są problemy związane z wyznaczaniem stanu spójnego w systemie w pełni asynchronicznym. Wreszcie omówione są algorytmy konstruujące spójny stan globalny (m.in. algorytm Lamporta dla systemu z kanałami FIFO oraz algorytm Lai-Yanga dla systemów z kanałami non-FIFO).

W dalszej części omawiane są zagadnienia związane z niezawodnością przetwarzania. Zdefiniowane są modele awarii, wprowadzona jest abstrakcja detektora błędów i omówione są różne rodzaje detektorów błędów oraz przykładowe modele ich realizacji. Pokazane są także sposoby realizacji abstrakcji łączy niezawodnych w oparciu o zawodne fizyczne łącza komunikacyjne oraz omówione są algorytmy realizujące mechanizmy niezawodnej komunikacji grupowej, wykorzystujące detektory błędów o różnych właściwościach.

Wykład prezentuje także problematykę związaną z osiągnięciem konsensusu w systemach rozproszonych. Pokazane jest, dlaczego w ogólności nie jest możliwe wyznaczenie konsensusu w systemie w pełni asynchronicznym, w którym istnieje możliwości awarii chociaż jednego procesu. Następnie przeanalizowane są algorytmy rozwiązujące konsensus przy pewnych dodatkowych założeniach (odnośnie dostępności detektorów błędów określonych klas).

Wykład przedstawia także zagadnienia związane z wyznaczaniem zakończenia w systemach rozproszonych. Wprowadzone są definicje (m.in. zakończenia statycznego i dynamicznego) oraz przedstawione są liczne algorytmy rozwiązujące problem zakończenia dla systemów o różnych topologiach i różnych modelach przetwarzania.

Dla większości przedstawianych algorytmów analizowana jest ich poprawność oraz złożoność czasowa i komunikacyjna.

Na laboratorium studenci zapoznają się z dwoma bibliotekami służącymi do implementacji aplikacji rozproszonych: bibliotekami PVM oraz MPI. Po zapoznaniu się z narzędziami, studenci konfrontują wiadomości uzyskane na wykładzie z praktyczną implementacją algorytmów rozproszonych. Demonstrowane są skutki braku pełnej synchronizacji zegarów nawet w środowisku węzłów połączonych szybką siecią lokalną. Przedstawione są skutki braku synchronizacji między procesami w środowisku rozproszonym. Następnie studenci implementują zegary logiczne, zrównoleglają proste program (np. łamiący zaszyfrowane hasło metodą brutalnej siły, oraz wyliczania liczby ? metodą Monte Carlo). Rozwiązywany jest także problem wyznaczania spójnego stanu globalnego.

Każde laboratorium składa się z przedstawienia problemu, wspólnej dyskusji w celu odnalezienia rozwiązania, a następnie implementacji. W dalszej części, w ramach uzupełnienia wykładu, przedstawione jest wprowadzenie do problemów zakleszczenia oraz wzajemnego wykluczania w kontekście systemów rozproszonych. Studenci następnie pracują w dwuosobowych zespołach. Otrzymują szczegółowe zagadnienia do rozwiązania, wraz z wskazówkami literaturowymi mogącymi pomóc im w zaprojektowaniu rozwiązania. Muszą utworzyć samodzielnie algorytm rozwiązujący rozwiązanie, lub dostosować jeden odnalezionych algorytmów. Przygotowane rozwiązanie musi zostać najpierw zaprojektowane i zaaprobowane, a dopiero po jego analizie zaimplementowane.

Cześć wymienionych wyżej treści programowych realizowana jest w ramach pracy własnej studenta.

Metody dydaktyczne:

1. wykład: prezentacja multimedialna, prezentacja ilustrowana przykładami podawanymi na tablicy, rozwiązywanie zadań, dyskusja
2. ćwiczenia laboratoryjne: rozwiązywanie zadań, ćwiczenia praktyczne, dyskusja, praca w zespole, pokaz multimedialny

Literatura podstawowa:

1. William Stallings, Cryptography and Network Security: Principles and Practice, Pearson Education, 2016

2. David Salomon, Elements of Computer Security, Springer-Verlag, 2010

3. Michał Szychowiak, Bezpieczeństwo systemów informatycznych. Zaawansowane ćwiczenia w systemach Windows i Linux, WPP, 2017

Literatura uzupełniająca:

1. Ross Anderson, Security Engineering, John Wiley & Sons, 2003 (http://www.cl.cam.ac.uk/~rja14/book.html)

2. Neil Smyth, Security+ Essentials, Payload Media, 2012 (http://techotopia.com/index.php?title=Security%2B_Essentials)

3. John Savard, A Cryptographic Compendium (http://www.quadibloc.com/crypto/jscrypt.htm)

4. Bartosz Brodecki, Jerzy Brzeziński, Piotr Sasak, Michał Szychowiak: Problemy bezpieczeństwa w architekturze SOA, w Damian Niemir, Maciej Stroiński, Jan Węglarz (Eds.): Nauka w obliczu społeczeństwa cyfrowego, Ośrodek Wydawnictw Naukowych, 2010, ISBN 978-83-7712-032-3, str. 233-246.

5. Michał Szychowiak: Bezpieczeństwo Systemów Informatycznych. http://wazniak.mimuw.edu.pl/index.php?title=Bezpieczeństwo_systemów_komputerowych

Bilans nakładu pracy przeciętnego studenta

Czynność	Czas (godz.)
1. udział w zajęciach laboratoryjnych:	16
2. przygotowanie merytoryczne przed zajęciami laboratoryjnymi:	16
3. udział w konsultacjach związanych z realizacją procesu kształcenia, w szczególności ćwiczeń laboratoryjnych / projektu	2
4. realizacja projektu, przygotowanie sprawozdania	16
5. obrona projektu (przedstawienie i omówienie propozycji rozwiązania, demonstracja działania gotowego rozwiązania, przedstawienie sprawozdania)	2
6. udział w wykładach	16
7. zapoznanie się ze wskazaną literaturą / materiałami dydaktycznymi (10 stron tekstu naukowego = 1 godz.), 200 stron	20
8. przygotowanie do kolokwium zaliczeniowego z wykładów	10

Obciążenie pracą studenta

forma aktywności	godzin	ECTS
Łączny nakład pracy	98	4
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	34	1
Zajęcia o charakterze praktycznym	50	2