**Nazwa przedmiotu:**

Analiza i projektowanie systemów informacyjnych

**Koordynator przedmiotu:**

Piotr Salata

**Status przedmiotu:**

Obowiązkowy

**Poziom kształcenia:**

Studia II stopnia

**Program:**

Informatyka

**Grupa przedmiotów:**

Przedmioty techniczne - zaawansowane

**Kod przedmiotu:**

APSI

**Semestr nominalny:**

2 / rok ak. 2015/2016

**Liczba punktów ECTS:**

5

**Liczba godzin pracy studenta związanych z osiągnięciem efektów uczenia się:**

30 godzin wykładu
30 godzin projektu
40 godzin samodzielnej realizacji projektu
20 godzin przygotowania do egzaminu
4 godziny egzaminu
w sumie 124 godziny, co daje ok. 5 ECTS

**Liczba punktów ECTS na zajęciach wymagających bezpośredniego udziału nauczycieli akademickich:**

30 godzin wykładu
30 godzin projektu
4 godziny egzaminu
w sumie 64 godziny, co daje nieco poniżej 2 ECTS

**Język prowadzenia zajęć:**

polski

**Liczba punktów ECTS, którą student uzyskuje w ramach zajęć o charakterze praktycznym:**

30 godzin projektu
40 godzin samodzielnej realizacji projektu
w sumie 70 godzin, co daje ok. 3 ECTS

**Formy zajęć i ich wymiar w semestrze:**

|  |  |
| --- | --- |
| Wykład:  | 30h |
| Ćwiczenia:  | 0h |
| Laboratorium:  | 0h |
| Projekt:  | 30h |
| Lekcje komputerowe:  | 0h |

**Wymagania wstępne:**

podstawowa znajomość inżynierii oprogramowania, technik modelowania: UML, E-R, zagadnień dotyczących analizy i projektowania baz danych

**Limit liczby studentów:**

60

**Cel przedmiotu:**

Celem wykładu jest zapoznanie słuchaczy z cyklem życia projektu informatycznego oraz z zagadnieniami dotyczącymi prac analitycznych i projektowych.
Przedstawione są tradycyjne i zwinne metody realizacji projektów, w szczególności metoda Scrum. Zasadniczym celem jest dogłębne zrozumienie przez słuchaczy problematyki prowadzenia projektów, występujących problemów, wad i zalet poszczególnych metod oraz zasad doboru metod i technik odpowiednich do realizowanych projektów.
W zakresie zagadnień analityczno-projektowych omawiane są metody i techniki związane ze zbieraniem i analizą wymagań, analizą systemową, definiowaniem architektury oraz projektem ogólnym systemu.
Celem zajęć projektowych jest praktyczna weryfikacja wiedzy nabytej na wykładzie oraz nabycie podstawowych umiejętności dotyczących prac analitycznych i projektowych.
Istotnym celem jest również nauka pracy zespołowej. Studenci realizują projekty w zespołach (standardowo 4-osobowych) i mają do wyboru możliwość realizacji projektu w trybie tradycyjnym lub w metodzie Scrum.

**Treści kształcenia:**

Wprowadzenie: elementy cyklu życia systemu informatycznego.
Model kaskadowy i jego problemy.
Metody rozwiązywania problemów modelu kaskadowego, technika kontroli zmian, model przyrostowy i jego problemy.
Modele iteracyjne, model spiralny: ich założenia, cechy i mechanizmy; technika Timeboxing. Modele hybrydowe: RUP.
Podsumowanie i porównanie modeli cyklu życia systemu.
Problemy tradycyjnego podejścia do realizacji projektu informatycznego, wprowadzenie do zasad podejścia zwinnego (Agile).
Manifest i pryncypia podejścia zwinnego. Korzyści i problemy wynikające ze stosowania metod zwinnych.
Analiza wymagań: wprowadzenie do technik zbierania i specyfikowania wymagań.
Technika przypadków użycia (Use Case): sposób opisu, różnicowanie przypadków użycia (biznesowe – systemowe – współpracy, wysokiego poziomu – rozszerzone, główne – drugorzędne, istotne – rzeczywiste, black box – white box). Metody identyfikacji przypadków użycia (3 perspektywy), rola przypadków użycia w procesie realizacji systemu.
Strukturalizacja modelu przypadków użycia: istota i techniki strukturalizacji, związki pomiędzy aktorami, związki pomiędzy przypadkami użycia. Analiza modelu.
Technika historyjek użytkownika (User Stories): istota, zasady i cechy techniki, rola historyjek użytkownika w procesie realizacji systemu, forma opisu - kanoniczna i rozszerzona, reguły INVEST i CCC, zasady i techniki dekompozycji historyjek.
Relacja pomiędzy przypadkami użycia, a historyjkami użytkownika. Stosowalność obu technik w projektach informatycznych.
Wymagania niefunkcjonalne: znaczenie dla projektu, sposób definiowania. Wymagania ilościowe i jakościowe.
Przegląd wymagań niefunkcjonalnych: niezawodność, dostępność, bezpieczeństwo, pojemność, wydajność, sprawność, efektywność, zarządzalność, wiarygodność, trwałość, użyteczność, ergonomia, zrozumiałość, wielojęzyczność, zgodność z normami, kompatybilność, topologia, modyfikowalność, indywidualizacja, uniwersalność, elastyczność, przenośność, kompletność, testowalność, reużywalność, skalowalność, rozszerzalność.
Architektura systemu: zakres, cel i zasady definiowania architektury. Sposób opisu i stosowane techniki.
Metoda Scrum: wprowadzenie do metody, podstawy pojęcia i techniki: Sprint (Sprint Plannig, Sprint Review, Sprint Retrospective), Backlog; uczestnicy procesu: zespół, Scrum Master, Product Owner. Zasady stosowania, zalety, wady, i ograniczenia metody, problemy we wdrażaniu, 3 „typy” Scrum.
Techniki prototypowania i projektowania: Spike, Tracer Bullet, Personas.
Analiza systemowa: model analityczny, techniki opisu (analiza obiektowa). Relacja model przypadków użycia – model analityczny. Zaawansowane zagadnienia modelowania obiektowego: semantyka diagramu klas, w tym 12 typów związków agregacji, semantyka diagramu stanów.
Projektowanie: cel zadania, zawartość i sposób opisu specyfikacji, stosowane techniki opisu specyfikacji.
Projekt ogólny systemu: przegląd zagadnień koniecznych do ujęcia w specyfikacji.

**Metody oceny:**

Ocena oparta jest na niezależnie ocenianej części projektowej i wykładowej (egzaminie).
W ramach projektu każdy z członków zespołu otrzymuje ocenę indywidualną w standardowej skali 2-5. Dla zaliczenia przedmiotu ocena z projektu musi być wyższa, niż 2.
Egzamin standardowo ma formę pisemną, jednakże gdy do danego terminu przystępuje nie więcej, niż 5 studentów, przeprowadzany jest w formie ustnej. Egzamin oceniany jest w zwykłej skali 2-5. Osoba, która nie przystąpi do egzaminu, otrzymuje ocenę 2. Przy zdawaniu w dwóch terminach liczona jest wyższa z uzyskanych ocen.
Ostateczna ocena z przedmiotu jest średnią arytmetyczną obu otrzymanych ocen zaokrągloną w górę do 0,5, z wyjątkiem sytuacji, gdy oceną z egzaminu jest 2 – wtedy ocena końcowa zaokrąglana jest w dół.

**Egzamin:**

tak

**Literatura:**

Robert Martin: Agile Software Development, Principles, Patterns, and Practices, Prentice Hall, 2002.
James Coplien, Neil Harrison: Organizational Patterns of Agile Software Development, Prentice Hall, 2004.
Alistair Cockburn: Agile Software Development: The Cooperative Game, Addison-Wesley, 2006.
Alistair Cockburn: Writing Effective Use Cases, Addison-Wesley, 2001 (WNT 2004).
Jim Highsmith: Agile Project Management, Addison-Wesley, 2004.
Mike Cohn: Succeeding With Agile: Software Development Using Scrum, Addison-Wesley, 2010.
Grady Booch, James Rumbaugh, Ivar Jacobson: The Unified Modeling Language User Guide (2nd edition), Addison-Wesley, 2005.
Ivar Jacobson, Grady Booch, James Rumbaugh: The Unified Software Development Process, Addison-Wesley, 1999.
Grady Booch, Robert Maksimchuk, Michael Engel, Bobbi Young, Jim Conallen, Kelli Houston: Object-Oriented Analysis And Design With Applications (3rd Edition), Addison-Wesley, 2007.
Kurt Bittner, Ian Spence: Use Case Modeling, Addison-Wesley, 2002.
Karl Wiegers: Software Requirements, Microsoft Press, 2003.
Dean Leffingwell, Don Widrig: Managing Software Requirements: A Use Case Approach, Addison-Wesley, 2003.
Jeffrey Whitten, Lonnie Bentley: Systems Analysis and Design Methods (7th Edition), McGraw Hill, 2007.
Jim Arlow, Ila Neustadt: UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design (2nd Edition), Addison-Wesley, 2005.
Jon Hunt: The Unified Process for Practitioners: Object Oriented Design, UML and Java, Sprinter-Verlag 2000.
Per Kroll, Philippe Krutchen: The Rational Unified Process Made Easy: A Practitioner's Guide to Rational Unified Process, Addison-Wesley, 2003 (WNT 2006).
Alan Dennis, Barbara Haley Wixom, David Tegarden: Systems Analysis and Design with UML Version 2.0: An Object-Oriented Approach, Wiley, 2005.
Ian Graham: Object-Oriented Methods: Principles & Practice, Addison-Wesley, 2000.
James J. Odell: Advanced Object-Oriented Analysis & Design Using UML, SIGS, 1998.
Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995.

**Witryna www przedmiotu:**

https://piotrsalata.pl/dydaktyka/apsi

**Uwagi:**

## Efekty przedmiotowe

### Profil ogólnoakademicki - wiedza

**Efekt APSI\_W01:**

zna modele cyklu życia systemu informatycznego, ich problemy, wady i zalety

Weryfikacja:

egzamin, projekt

**Powiązane efekty kierunkowe:** K\_W12, K\_W07, K\_W08

**Powiązane efekty obszarowe:** T2A\_W08, T2A\_W05, T2A\_W07

**Efekt APSI\_W02:**

zna techniki zbierania i analizy wymagań, w szczególności techniki przypadków użycia (Use Case) i historyjek użytkownika (User Stories)

Weryfikacja:

egzamin, projekt

**Powiązane efekty kierunkowe:** K\_W05, K\_W08

**Powiązane efekty obszarowe:** T2A\_W04, T2A\_W07

**Efekt APSI\_W03:**

zna techniki dotyczące analizy systemowej, zna semantykę modelu UML

Weryfikacja:

egzamin, projekt

**Powiązane efekty kierunkowe:** K\_W02, K\_W05, K\_W08

**Powiązane efekty obszarowe:** T2A\_W03, T2A\_W04, T2A\_W07

**Efekt APSI\_W05:**

zna metodę Scrum

Weryfikacja:

egzamin, projekt (opcjonalnie)

**Powiązane efekty kierunkowe:** K\_W08

**Powiązane efekty obszarowe:** T2A\_W07

**Efekt APSI\_W04:**

zna zagadnienia dotyczące specyfikacji projektu ogólnego systemu informatycznego

Weryfikacja:

egzamin, projekt

**Powiązane efekty kierunkowe:** K\_W05, K\_W08

**Powiązane efekty obszarowe:** T2A\_W04, T2A\_W07

### Profil ogólnoakademicki - umiejętności

**Efekt APSI\_U01:**

potrafi dobrać odpowiednią metodę i techniki do konkretnego charakteru realizowanego projektu informatycznego

Weryfikacja:

projekt

**Powiązane efekty kierunkowe:** K\_U13

**Powiązane efekty obszarowe:** T2A\_U18

**Efekt APSI\_U02:**

potrafi zebrać wymagania na system informatyczny i dokonać ich analizy wykorzystując odpowiednie techniki

Weryfikacja:

projekt

**Powiązane efekty kierunkowe:** K\_U02, K\_U05, K\_U10, K\_U12, K\_U13

**Powiązane efekty obszarowe:** T2A\_U02, T2A\_U07, T2A\_U09, T2A\_U15, T2A\_U17, T2A\_U18

**Efekt APSI\_U03:**

potrafi zdefiniować architekturę systemu informatycznego stosując odpowiednie metody weryfikacji jej poprawności, w tym prototypowanie

Weryfikacja:

projekt

**Powiązane efekty kierunkowe:** K\_U08, K\_U09, K\_U10, K\_U12, K\_U13, K\_U14

**Powiązane efekty obszarowe:** T2A\_U12, T2A\_U11, T2A\_U15, T2A\_U17, T2A\_U18, T2A\_U19

**Efekt APSI\_U04:**

potrafi opracować specyfikację analityczną systemu informatycznego

Weryfikacja:

projekt

**Powiązane efekty kierunkowe:** K\_U05, K\_U12, K\_U13

**Powiązane efekty obszarowe:** T2A\_U07, T2A\_U09, T2A\_U17, T2A\_U18

**Efekt APSI\_U05:**

potrafi opracować specyfikację ogólnego projektu systemu informatycznego oraz dobrać odpowiednie rozwiązania techniczne

Weryfikacja:

projekt

**Powiązane efekty kierunkowe:** K\_U05, K\_U08, K\_U10, K\_U12, K\_U13, K\_U14

**Powiązane efekty obszarowe:** T2A\_U07, T2A\_U09, T2A\_U12, T2A\_U15, T2A\_U17, T2A\_U18, T2A\_U19

### Profil ogólnoakademicki - kompetencje społeczne

**Efekt APSI\_K01:**

potrafi pracować zespołowo

Weryfikacja:

projekt

**Powiązane efekty kierunkowe:** K\_K01

**Powiązane efekty obszarowe:** T2A\_K06